Journal of Big Data

CASE STUDY Open Access

A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions

Bharti Khemani[1], Shruti Patil[2*], Ketan Kotecha[2] and Sudeep Tanwar[3]

*Correspondence:
shruti.patil@sitpune.edu.in

[1] Symbiosis Institute of Technology Pune Campus, Symbiosis International (Deemed University) (SIU), Lavale, Pune 412115, India
[2] Symbiosis Centre for Applied Artificial Intelligence (SCAAI), Symbiosis Institute of Technology Pune Campus, Symbiosis International (Deemed University) (SIU), Lavale, Pune 412115, India
[3] IEEE, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad, India

Abstract

Deep learning has seen significant growth recently and is now applied to a wide range of conventional use cases, including graphs. Graph data provides relational information between elements and is a standard data format for various machine learning and deep learning tasks. Models that can learn from such inputs are essential for working with graph data effectively. This paper identifies nodes and edges within specific applications, such as text, entities, and relations, to create graph structures. Different applications may require various graph neural network (GNN) models. GNNs facilitate the exchange of information between nodes in a graph, enabling them to understand dependencies within the nodes and edges. The paper delves into specific GNN models like graph convolution networks (GCNs), GraphSAGE, and graph attention networks (GATs), which are widely used in various applications today. It also discusses the message-passing mechanism employed by GNN models and examines the strengths and limitations of these models in different domains. Furthermore, the paper explores the diverse applications of GNNs, the datasets commonly used with them, and the Python libraries that support GNN models. It offers an extensive overview of the landscape of GNN research and its practical implementations.

Keywords: Graph Neural Network (GNN), Graph Convolution Network (GCN), GraphSAGE, Graph Attention Networks (GAT), Message Passing Mechanism, Natural Language Processing (NLP)

Introduction

Graph Neural Networks (GNNs) have emerged as a transformative paradigm in machine learning and artificial intelligence. The ubiquitous presence of interconnected data in various domains, from social networks and biology to recommendation systems and cybersecurity, has fueled the rapid evolution of GNNs. These networks have displayed remarkable capabilities in modeling and understanding complex relationships, making them pivotal in solving real-world problems that traditional machine-learning models struggle to address. GNNs' unique ability to capture intricate structural information inherent in graph-structured data is significant. This information often manifests as dependencies, connections, and contextual relationships essential for making informed

predictions and decisions. Consequently, GNNs have been adopted and extended across various applications, redefining what is possible in machine learning.

In this comprehensive review, we embark on a journey through the multifaceted landscape of Graph Neural Networks, encompassing an array of critical aspects. Our study is motivated by the ever-increasing literature and diverse perspectives within the field. We aim to provide researchers, practitioners, and students with a holistic understanding of GNNs, serving as an invaluable resource to navigate the intricacies of this dynamic field. The scope of this review is extensive, covering fundamental concepts that underlie GNNs, various architectural designs, techniques for training and inference, prevalent challenges and limitations, the diversity of datasets utilized, and practical applications spanning a myriad of domains. Furthermore, we delve into the intriguing future directions that GNN research will likely explore, shedding light on the exciting possibilities.

In recent years, deep learning (DL) has been called the gold standard in machine learning (ML). It has also steadily evolved into the most widely used computational technique in ML, producing excellent results on various challenging cognitive tasks, sometimes even matching or outperforming human ability. One benefit of DL is its capacity to learn enormous amounts of data [1]. GNN variations such as graph convolutional networks (GCNs), graph attention networks (GATs), and GraphSAGE have shown groundbreaking performance on various deep learning tasks in recent years [2].

A graph is a data structure that consists of nodes (also called vertices) and edges. Mathematically, it is defined as $G = (V, E)$, where V denotes the nodes and E denotes the edges. Edges in a graph can be directed or undirected based on whether directional dependencies exist between nodes. A graph can represent various data structures, such as social networks, knowledge graphs, and protein–protein interaction networks. Graphs are non-Euclidean spaces, meaning that the distance between two nodes in a graph is not necessarily equal to the distance between their coordinates in an Euclidean space. This makes applying traditional neural networks to graph data difficult, as they are typically designed for Euclidean data.

Graph neural networks (GNNs) are a type of deep learning model that can be used to learn from graph data. GNNs use a message-passing mechanism to aggregate information from neighboring nodes, allowing them to capture the complex relationships in graphs. GNNs are effective for various tasks, including node classification, link prediction, and clustering.

Organization of paper

The paper is organized as follows:

1) The primary focus of this research is to comprehensively examine Concepts, Architectures, Techniques, Challenges, Datasets, Applications, and Future Directions within the realm of Graph Neural Networks.
2) The paper delves into the Evolution and Motivation behind the development of Graph Neural Networks, including an analysis of the growth of publication counts over the years.
3) It provides an in-depth exploration of the Message Passing Mechanism used in Graph Neural Networks.

4) The study presents a concise summary of GNN learning styles and GNN models, complemented by an extensive literature review.
5) The paper thoroughly analyzes the Advantages and Limitations of GNN models when applied to various domains.
6) It offers a comprehensive overview of GNN applications, the datasets commonly used with GNNs, and the array of Python libraries that support GNN models.
7) In addition, the research identifies and addresses specific research gaps, outlining potential future directions in the field.

"Introduction" section describes the Introduction to GNN. "Background study" section provides background details in terms of the Evolution of GNN. "Research motivation" section describes the research motivation behind GNN. Section IV describes the GNN message-passing mechanism and the detailed description of GNN with its Structure, Learning Styles, and Types of tasks. "GNN Models and Comparative Analysis of GNN Models" section describes the GNN models with their literature review details and comparative study of different GNN models. "Graph Neural Network Applications" section describes the application of GNN. And finally, future direction and conclusions are defined in "Future Directions of Graph Neural Network" and "Conclusions" sections, respectively. Figure 1 gives the overall structure of the paper.

Background study

As shown in Fig. 2 below, the evolution of GNNs started in 2005. For the past 5 years, research in this area has been going into great detail. Neural graph networks are being used by practically all researchers in fields such as NLP, computer vision, and healthcare.

Graph neural network research evolution

Graph neural networks (GNNs) were first proposed in 2005, but only recently have they begun to gain traction. GNNs were first introduced by Gori [2005] and Scarselli [2004, 2009]. A node's attributes and connected nodes in the graph serve as its natural definitions. A GNN aims to learn a state embedding $h_v \in R^s$ that encapsulates each node's neighborhood data. The distribution of the expected node label is one example of the output. An s-dimension vector of node v, the state embedding h_v, can be utilized to generate an output O_v, such as the anticipated distribution node name. The predicted node label (O_v) distribution is created using the state embedding h_v [30]. Thomas Kipf and Max Welling introduced the convolutional graph network (GCN) in 2017. A GCN layer defines a localized spectral filter's first-order approximation on graphs. GCNs can be thought of as convolutional neural networks that have been expanded to handle graph-structured data.

Graph neural network evolution

As shown in Fig. 3 below, research on graph neural networks (GNNs) began in 2005 and is still ongoing. GNNs can define a broader class of graphs that can be used for node-focused tasks, edge-focused tasks, graph-focused tasks, and many other applications. In 2005, Marco Gori introduced the concept of GNNs and defined recursive

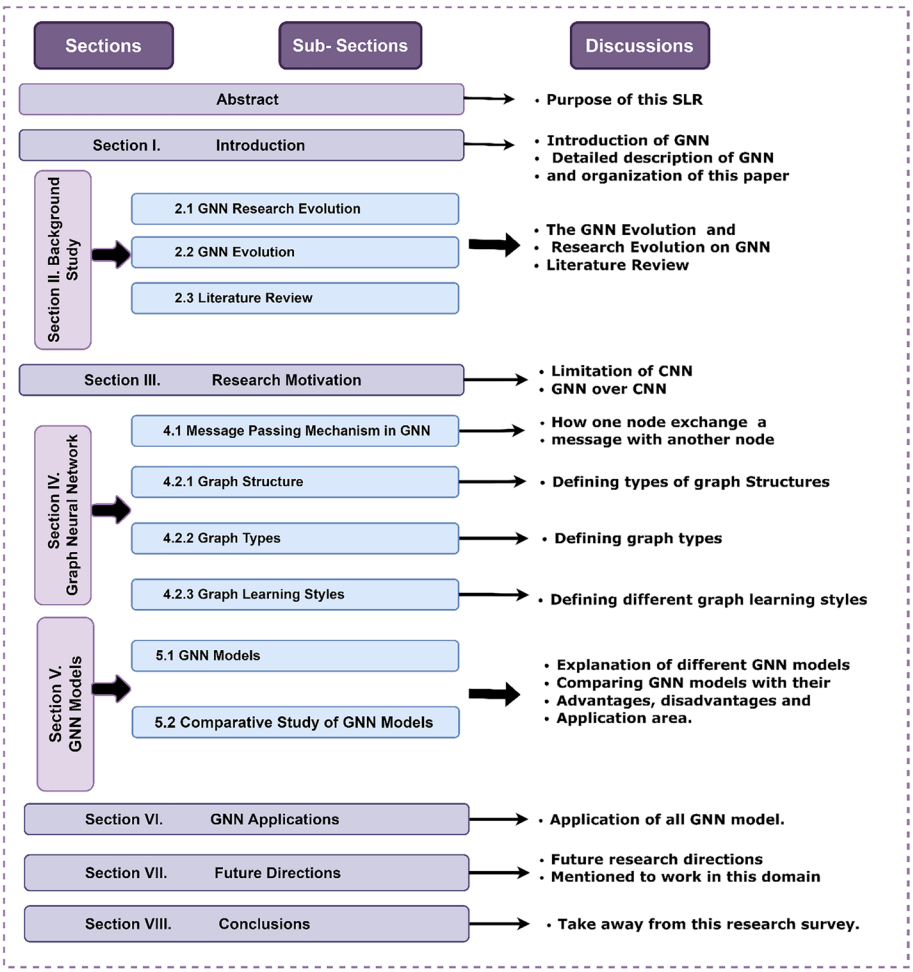

Fig. 1 The overall structure of the paper

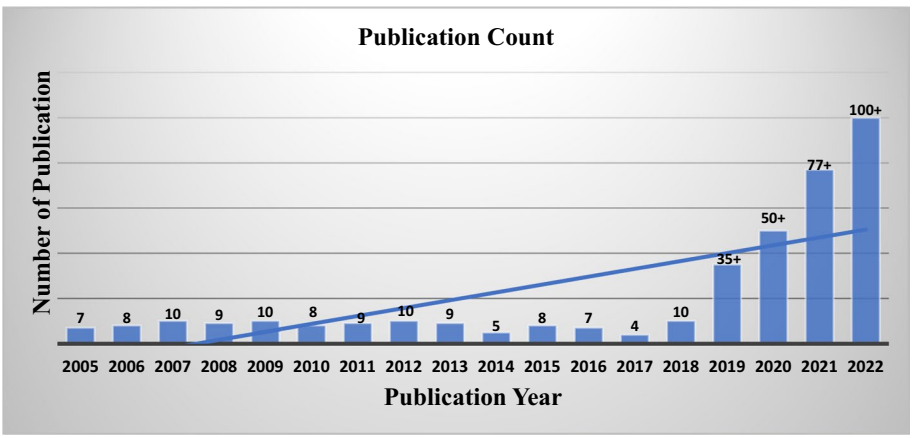

Fig. 2 Year-wise publication count of GNN (2005–2022)

neural networks extended by GNNs [4]. Franco Scarselli also explained the concepts for ranking web pages with the help of GNNs in 2005 [5]. In 2006, Swapnil Gandhi and Anand Padmanabha Iyer of Microsoft Research introduced distributed deep

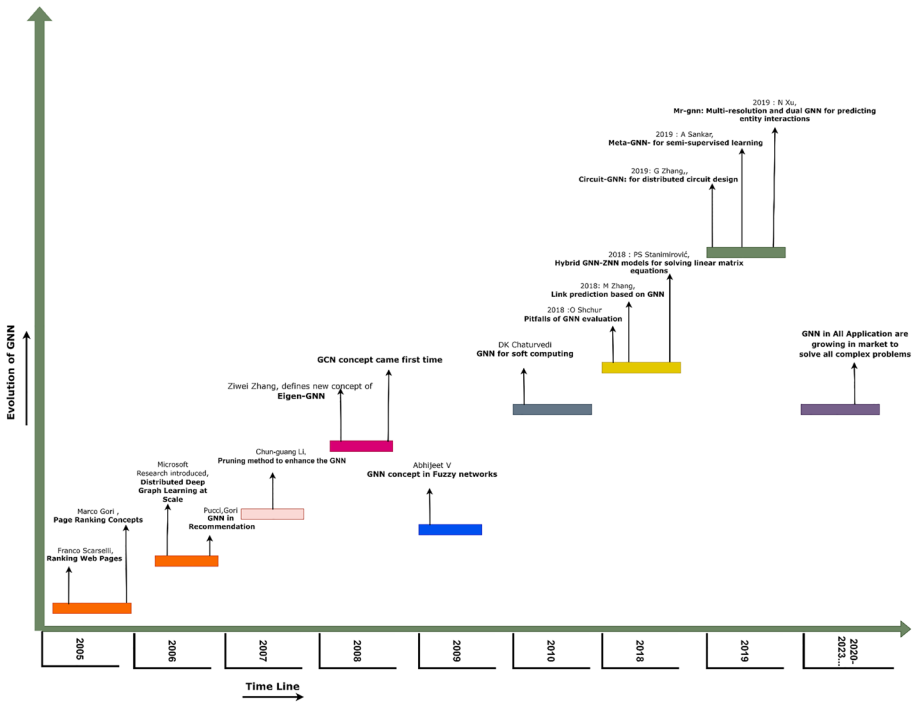

Fig. 3 Graph Neural Network Evolution

graph learning at scale, which defines a deep graph neural network [6]. They explained new concepts such as GCN, GAT, etc. [1]. Pucci and Gori used GNN concepts in the recommendation system.

2007 Chun Guang Li, Jun Guo, and Hong-gang Zhang used a semi-supervised learning concept with GNNs [7]. They proposed a pruning method to enhance the basic GNN to resolve the problem of choosing the neighborhood scale parameter. In 2008, Ziwei Zhang introduced a new concept of Eigen-GNN [8], which works well with several GNN models. In 2009, Abhijeet V introduced the GNN concept in fuzzy networks [9], proposing a granular reflex fuzzy min–max neural network for classification. In 2010, DK Chaturvedi explained the concept of GNN for soft computing techniques [10]. Also, in 2010, GNNs were widely used in many applications. In 2010, Tanzima Hashem discussed privacy-preserving group nearest neighbor queries [11]. The first initiative to use GNNs for knowledge graph embedding is R-GCN, which suggests a relation-specific transformation in the message-passing phases to deal with various relations.

Similarly, from 2011 to 2017, all authors surveyed a new concept of GNNs, and the survey linearly increased from 2018 onwards. Our paper shows that GNN models such as GCN, GAT, RGCN, and so on are helpful [12].

Literature review

In the Table 1 describe the literature survey on graph neural networks, including the application area, the data set used, the model applied, and performance evaluation. The literature is from the years 2018 to 2023.

Table 1 GNN papers with their performance

Refs.	Application area	Dataset used	Model applied	Summary	Performance evaluation
[13] 2018	Link prediction	Usair NS PB Yeast C.ele Power Router And E.coli	GNN	They defend the use of prediction heuristics to learn from local enclosing subgraphs	Area Under Curve: Usair—97.09 ± 0.70 NS—97.71 ± 0.93 PB—95.01 ± 0.34 Yeast—97.20 ± 0.64 C.ele—89.54 ± 2.04 Power—84.18 ± 1.82 Router—95.68 ± 1.22 E.coli—97.22 ± 0.28
[14] 2018	Solving matrix equations	Own examples	Hybrid GNN GNN + ZNN (Zhang Neural Network)	They solve the matrix equations BX = D and XC = D in time-invariant cases	The global convergence rate has improved by taking their example
[15] 2018	Solving matrix equations	Explains theorems with their examples	Gradient-based neural dynamics (GND)	Solve matrix equation AXB = D	The global convergence rate has improved by taking their example
[16] 2019	Link forecast Recommendation Node Clustering and Node Classification	Academic I (A-I) Academic II (A-II) Movies Review (R-I) Cds Review	Hetgnn	Hetgnn considered combining heterogeneous types, type-based neighbors, and heterogeneous node contents	AUC: Multi-label classification—0.978 Node clustering – 0.901
[17] 2019	Social Recommendation	Ciao and Epinions	Graphrec	They predict ratings and provide interactions and opinions on the user-item graph	RMSE: Ciao—0.9794 Epinions—0.8168
[18] 2019	Chinese-named entity recognition	Ontonotes MSRA Weibo Resume	Lexicon-based GNN	Chinese NER is achieved as a graph node classification using a vocabulary to build a graph neural network	74.89 93.46 60.21 95.37
[19] 2019	Taxable detection & structure recognition	UW3, UNLV, ICDAR 2013	CNN + GNN	The best networks for detecting representative visual features are convolutional neural networks, whereas the best networks for quick message transfer between vertices are graph networks. With the help of the gather operation, we have demonstrated how to integrate these two skills	68.5

Table 1 (continued)

Refs.	Application area	Dataset used	Model applied	Summary	Performance evaluation
[20] 2019	Link prediction Pair-wise node classification	Grid Communities PPI	P-GNN Point of view GNN	To compute node embeddings that contain node positional information while maintaining inductive capability and leveraging node attributes, they introduce a new class of GNN	AUC: 0.940 ± 0.027 0.985 ± 0.008 0.808 ± 0.003
[21] 2020	Time-series Forecasting	METR-LA PEMS-BAY PEMS07 PEMS03 PEMS04 PEMS08 Solar Electricity ECG5000 COVID-19	Spectral Temporal GNN Stemgnn)		RMSE: 5.06 2.48 4.01 21.64 32.15 24.93 0.07 0.06 0.07 19.3
[22] [2020]	Citation Network	Cora, Citeseer, Pubmed, and NELL	Continuous GNN(CGNN)	Enable continuous instances to be handled by existing discrete graph neural networks by describing the evolution of node representations with ODE	82.1 ± 1.3 72.9 ± 0.9 82.7 ± 1.4 73.1 ± 0.9
[23] 2020	Node representation visualization	Cora, Citeseer, Pubmed Coauthors	Differentiable group normalization (DGN), simple graph convolution networks (SGC)	They propose group distance ratio and instance information gain as two over-smoothing metrics based on graph architectures	80.2% 58.2% 76.2% 85.8%
[24] 2020	Medical	MUTAG PTC COX2 PROTEINS NCI1	Implicit graph neural network (IGCN)	They outline a Perron-Frobenius hypothesis necessary condition for very well and a projected gradient descent training approach	89.3 ± 6.7 70.1 ± 5.6 86.9 ± 4.0 77.7 ± 3.4 80.5 ± 1.9
[25] 2021	Text classification	IMDB webkb R52 R8 AG_news	Deep Attention Diffusion Graph Neural Network (DADGNN)	Proposes an attention diffusion technique that captures non-direct-neighbor context information in a single layer and decouples the required GNN training processes (representation transformation and propagation)	88.49 ± 0.59 90.92 ± 0.42 95.16 ± 0.22 98.15 ± 0.16 92.24 ± 0.36

Table 1 (continued)

Refs.	Application area	Dataset used	Model applied	Summary	Performance evaluation
[26] 2021	Medicines	COLL, MD17, and OC20	Neural Network For Geometric Messages Passing	Gemnet uses effective bilinear layers and symmetric message passing	34%, 41%, and 20%
[27] 2022	Feature extraction	Pavia University Salinas Houston 2013	Deep Hybrid Multi-Graph Neural Network (DHMG)	To reduce the noise in the graph, they created a unique ARMA filter and implemented it recursively	97.81 ± 0.82 98.33 ± 0.28 93.31 ± 0.65
[28] (2023)	Traffic Prediction	AIS data and global port geospatial data	GAT	Research on Multi-Port Ship Traffic Prediction Method Based on Spatiotemporal Graph Neural Networks	Around 90%
[29] (2023)	Traffic Forecasting	PEMS03, PEMS04, PEMS07, PEMS08	GNN	Hybrid GCN and branch-and-bound optimization for traffic flow forecasting	0.58 0.63 0.63 0.73

Research motivation

We employ grid data structures for normalization of image inputs, typically using an n*n-sized filter. The result is computed by applying an aggregation or maximum function. This process works effectively due to the inherent fixed structure of images. We position the grid over the image, move the filter across it, and derive the output vector as depicted on the left side of Fig. 4. In contrast, this approach is unsuitable when working with graphs. Graphs lack a predefined structure for data storage, and there is no inherent knowledge of node-to-neighbor relationships, as illustrated on the right side of Fig. 4. To overcome this limitation, we focus on graph convolution.

In the context of GCNs, convolutional operations are adapted to handle graphs' irregular and non-grid-like structures. These operations typically involve aggregating information from neighboring nodes to update the features of a central node. CNNs are primarily used for grid-like data structures, such as images. They are well-suited for tasks where spatial relationships between neighboring elements are crucial, as in image processing. CNNs use convolutional layers to scan small local receptive fields and learn hierarchical representations. GNNs are designed for graph-structured data, where edges connect entities (nodes). Graphs can represent various relationships, such as social networks, citation networks, or molecular structures. GNNs perform operations that aggregate information from neighboring nodes to update the features of a central node. CNNs excel in processing grid-like data with spatial dependencies; GNNs are designed to handle graph-structured data with complex relationships and dependencies between entities.

Limitation of CNN over GNN

Graph Neural Networks (GNNs) draw inspiration from Convolutional Neural Networks (CNNs). Before delving into the intricacies of GNNs, it is essential to understand why Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) may not suffice for effectively handling data structured as graphs. As illustrated in Fig. 5, Convolutional Neural Networks (CNNs) are designed for data that exhibits a grid structure, such as images. Conversely, Recurrent Neural Networks (RNNs) are tailored to sequences, like text.

Typically, we use arrays for storage when working with text data. Likewise, for image data, matrices are the preferred choice. However, as depicted in Fig. 5, arrays and matrices fall short when dealing with graph data. In the case of graphs, we require

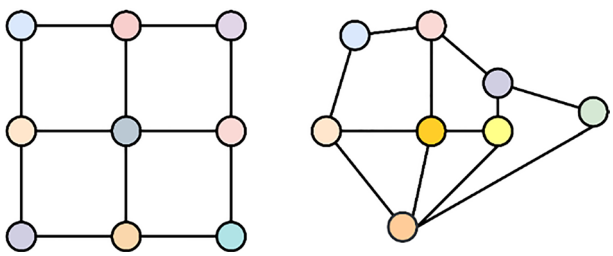

Fig. 4 CNN In Euclidean Space (Left), GNN In Euclidean Space (Right)

Table 2 Summary of Graph Convolution Network with the technique used, datasets used, and performance measure (accuracy)

Refs.	Application area	Method applied	Dataset	A model with several layers and an activation Function	Accuracy
[33] (2016)	Node Classification	Graph Convolution Network	CORA	GCN model with 2 layers ReLU function	82.98
[33] (2017)	Semi-Supervised Node Classification	Graph Convolution Network	Zachary's Karate Club	GCN model with 2 layers ReLU function	90%
[33] (2016)	Semi-Supervised Node Classification	Graph Convolution Network	CORA	GCN model with 2 layers ReLU function	81.5%
[39] (2019)	Text Classification	GCN for Text Classification	20NG Ohsumed R52 R8 MR	2 Layer GCN ReLU function	0.8634 ± 0.0009 0.9707 ± 0.0010 0.9356 ± 0.0018 0.6836 ± 0.0056 0.7674 ± 0.0020
[31] (2019)	Node Classification	Node Classification	Cora, Citeseer, Pubmed Reddit	4-layer GCN ReLU function	74.60% 61.40% 86.20% 50.51%
[40] (2018)	Quiz	Question Answering by Reasoning	WIKIHOP	Two layers MLP	65.3% to 68.7%
[41] (2018)	Node Classification	Node Classification	Cora, Citeseer, Pubmed	2 Layer GCN ReLU function	70.3% 81.5% 79.0%
[42] (2019)	Node Classification	Hierarchical graph convolutional networks for semi-supervised node classification	Cora, Citeseer, Pubmed NELL	2 Layer GCN ReLU function	70.3% 81.5% 79.0% 73.0%
[43] (2020)	Traffic prediction	Traffic prediction	Real-time dataset	Message passing technique + Graph Convolution Network	70–75%
[44] (2023)	Motion Capture for Sporting Events	Graph Convolutional Neural Networks and Single Target Pose Estimation	COCO dataset	Graph Neural Network Combined With High Resolution Network (HRNET)	79.3
[45] (2023)	Defect Recognition	Deep Graph Convolutional Neural Network	9 different dataset	Graph Convolutional Neural Network (GCNN)	Around 90%
[46] (2023)	Flow Prediction	Graph Convolutional Long Short-Term Memory Neural Network Model	Société de transport de Laval (STL)	MLP, CNN, LSTM, BNG-ConvLSTM = bus network graph convolutional long short-term memory	73.3 70.0 80.2 85.3

a specialized technique known as Graph Convolution. This approach enables deep neural networks to handle graph-structured data directly, leading to a graph neural network.

Fig. 5 illustrates that we can employ masking techniques and apply filtering operations to transform the data into vector form when we have images. Conversely, traditional

Table 3 Summary of Graph Attention Network with Application area, technique, datasets used, and performance measure (accuracy)

Refs.	Application area	Method applied	Dataset	Layer size and activation function	Performance evaluation
[47] (2017)	Node Classification	Graph Attention Network (GAT)	CORA	GAT Method with 3 layers ReLU function	76.5%
[48] (2017)	Traffic prediction	Gated Residual Recurrent Graph Neural Networks	Citation	×	77.8%
[49] (2021)	Edge Detection	Sparse Graph Attention Network (GAT)	CORA	GAT Method with 1 layer ReLU function	82.5%
[50] (2021)	Fault Diagnosis	KNN + GAT	hardware-in-the-loop (HIL)	×	87.7%
[47] (2017)	Citation Network Node Classification	GAT	Cora Citeseer PubMed	GAT 64 hidden features (using ReLU)	F1- score $83.0 \pm 0.7\%$ $72.5 \pm 0.7\%$ $79.0 \pm 0.3\%$
[51] (2021)	Node-Prediction	GAT GAT- v2	OGB	LeakyReLU activation function	GAT 78.1 ± 0.59 GATv2 78.5 ± 0.38
[52] (2019)	Node Embeddings	Signed Graph Attention Network (Si-GAT)	Epinions	LeakyReLU	0.9293
[53] (2019)	Node Classification Task	Heterogeneous Graph Attention Network	IMDB DBLP ACM	Random walk-based methods	10.01 84.76 64.39
[54] (2021)	Node Classification Task	Hyperbolic Graph Attention Network	Cora Citeseer PubMed Amazon Photo	8, 16, 32, 64 (i.e., the number of hidden units in GNN	0.567 0.427 0.359 0.667
[55] (2023)	Rumor Detection	GAT and GRU	Weibo and Pheme dataset	Two-layer GAT having 4 attention-head to each layer	97.2%
[56] (2022)	Disease Prediction	Knowledge Graph Attention Network	Own dataset	fivefold cross validation with KGAT	84.76

masking methods are not applicable when dealing with graph data as input, as shown in the right image.

Graph neural network

Graph Neural Networks, or GNNs, are a class of neural networks tailored for handling data organized in graph structures. Graphs are mathematical representations of nodes connected by edges, making them ideal for modeling relationships and dependencies in complex systems. GNNs have the inherent ability to learn and reason about graph-structured data, enabling diverse applications. In this section, we first explained the passing mechanism of GNN ("Message Passing Mechanism in Graph Neural Network Section"), then described graphs related to the structure of graphs, graph types, and graph learning styles ("Description of GNN Taxonomy" Section).

Message passing mechanism in graph neural network

Graph symmetries are maintained using a GNN, an optimizable transformation on all graph properties (nodes, edges, and global context) (permutation invariances). Because

Table 4 Summary of GraphSAGE Network with Application area, technique, datasets used, and performance measure (accuracy)

Refs.	Application Area	Method Applied	Dataset	Accuracy
[48] (2017)	Citation Network	GraphSAGE Mean- aggregator	Citation	77.8%
		GraphSAGE-LSTM aggregator	Citation	78.8%
		GraphSAGE-pool aggregator	Citation	79.8%
[31] (2019)	Node Classification	4-layer GCN	Cora, Citeseer, PubMed Reddit	32.20% 53.60% 47.90% 96.47
[57] (2019)	Social Network Analysis Based on Graph SAGE	GraphSAGE (GCN)	microblogs	53.87%
[58] (2021)	Intrusion Detection	E-GraphSAGE E-ResGAT	UNSW-NB15 CIC-DarkNet CSE-CIC-IDS ToN-IoT	0.9868 0.8093 0.8774 0.9384
[59] (2019)	Data-Driven Node Sampling	GraphSAGE	PPI Reddit PubMed	0.813 0.954 0.898
[60] (2023)	Underwater Moving Object Detection	GraphSAGE + Aggregator(Mean, Max and LSTM)	Fish4Knowledge dataset	Mean: 98.51% Max: 94.46% LSTM: 98.50%

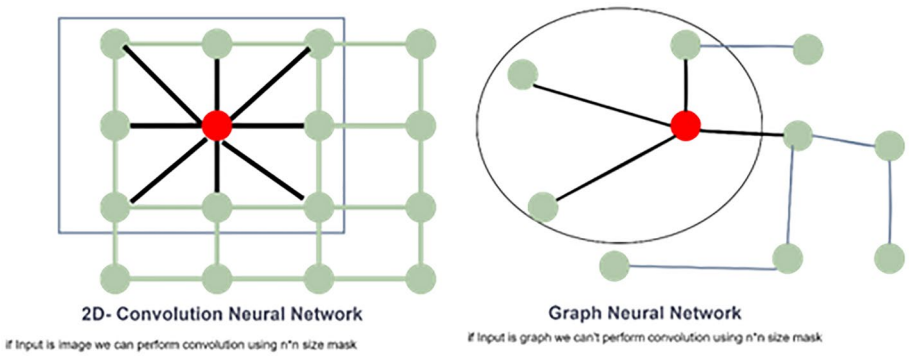

Fig. 5 Convolution can be performed if the input is an image using an n*n mask (Left). Convolution can't be achieved if the input is a graph using an n*n mask. (Right)

a GNN does not alter the connectivity of the input graph, the output may be characterized using the same adjacency list and feature vector count as the input graph. However, the output graph has updated embeddings because the GNN modified each node, edge, and global-context representation.

In Fig. 6, circles are nodes, and empty boxes show aggregation of neighbor/adjacent nodes. The model aggregates messages from A's local graph neighbors (i.e., B, C, and D). In turn, the messages coming from neighbors are based on information aggregated from their respective neighborhoods, and so on. This visualization shows a two-layer version of a message-passing model. Notice that the computation graph of the GNN forms a tree structure by unfolding the neighborhood around the target node [17]. Graph neural networks (GNNs) are neural models that capture the dependence of graphs via message passing between the nodes of graphs [30].

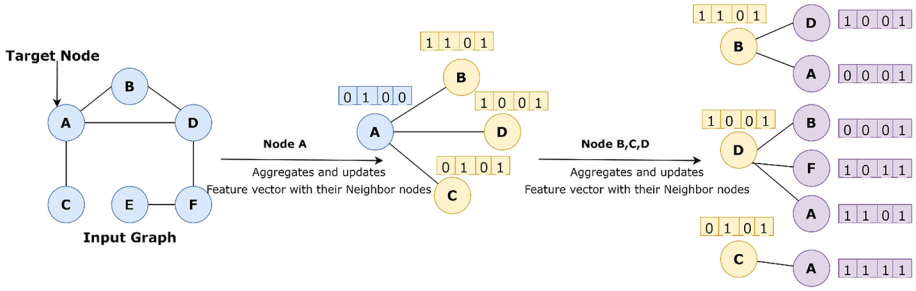

Fig. 6 How a single node aggregates messages from its adjacent neighbor nodes

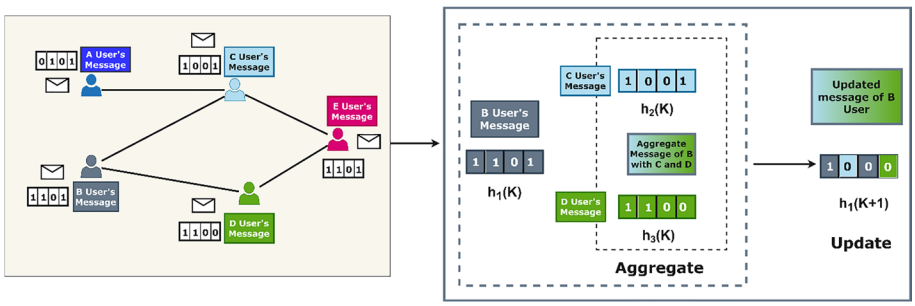

Fig. 7 Message passing mechanism in GNN

The message-passing mechanism of Graph Neural Networks is shown in Fig. 7. In this, we take an input graph with a set of node features X ε $R^{d \rightarrow} |V|$ and Use this knowledge to produce node embeddings z_u. However, we will also review how the GNN framework may embed subgraphs and whole graphs.

At each iteration, each node collects information from the neighborhood around it. Each node embedding has more data from distant reaches of the graph as these iterations progress. After the first iteration (k = 1), each node embedding expressly retains information from its 1-hop neighborhood, which may be accessed via a path in the length graph 1. [31]. After the second iteration (k = 2), each node embedding contains data from its 2-hop neighborhood; generally, after k iterations, each node embedding includes data from its k-hop setting. The kind of "information" this message passes consists of two main parts: structural information about the graph (i.e., degree of nodes, etc.), and the other is feature-based.

In the message-passing mechanism of a neural network, each node has its message stored in the form of feature vectors, and each time, the neighbor updates the information in the form of the feature vector [1]. This process aggregates the information, which means the grey node is connected to the blue node. Both features are aggregated and form new feature vectors by updating the values to include the new message.

$$h_u^{(k+1)} = UPDATE^{(k)}\left(h_u^{(k)}, AGGREGATE^{(k)}\left(\left\{h_v^{(k)}, \forall_v \in N(u)\right\}\right)\right) \tag{4.1}$$

$$= UPDATE^{(k)}\left(h_u^{(k)}, m_{N(u)}^{(k)}\right) \tag{4.2}$$

Equations 4.1 and 4.2 shows that h denotes the message, u represents the node number, and k indicates the iteration number. Where AGGREGATE and UPDATE are arbitrarily differentiable functions (i.e., neural networks), and mN(u) is the "message," which is aggregated from u's graph neighborhood N(u). We employ superscripts to identify the embeddings and functions at various message-passing iterations. The AGGREGATE function receives as input the set of embeddings of the nodes in the u's graph neighborhood N (u) at each iteration k of the GNN and generates a message. $m^k_{N(u)}$. Based on this aggregated neighborhood information. The update function first UPDATES the message and then combines the message.$m^k_{N(u)}$ with the previous message $h^{(k-1)}_u$ of node, u to generate the updated message h^k_u.

Description of GNN taxonomy

We can see from Fig. 8 below shows that we have divided our GNN taxonomy into 3 parts [30].

1. Graph Structures 2. Graph Types 3. Graph Learning Tasks

Graph structure

The two scenarios shown in Fig. 9 typically present are structural and non-structural. Applications involving molecular and physical systems, knowledge graphs, and other objects explicitly state the graph structure in structural contexts.

Graphs are implicit in non-structural situations. As a result, we must first construct the graph from the current task. For text, we must build a fully connected "a word" graph and a scene graph for images.

Graph types

There may be more information about nodes and links in complex graph types. Graphs are typically divided into 5 categories, as shown in Fig. 10.

Directed/undirected graphs A directed graph is characterized by edges with a specific direction, indicating the flow from one node to another. Conversely, in an undirected graph, the edges lack a designated direction, allowing nodes to interact bidirectionally. As illustrated in Fig. 11 (left side), the directed graph exhibits directed edges, while in Fig. 11 (right side), the undirected graph conspicuously lacks directional edges. In undirected graphs, it's important to note that each edge can be considered to comprise two directed edges, allowing for mutual interaction between connected nodes.

Static/dynamic graphs The term "dynamic graph" pertains to a graph in which the properties or structure of the graph change with time. In dynamic graphs shown in Fig. 12, it is essential to account for the temporal dimension appropriately. These dynamic graphs represent time-dependent events, such as the addition and removal of nodes and edges, typically presented as an ordered sequence or an asynchronous stream.

A noteworthy example of a dynamic graph can be observed in social networks like Twitter. In such networks, a new node is created each time a new user joins, and when a user follows another individual, a following edge is established. Furthermore, when users update their profiles, the respective nodes are also modified, reflecting the

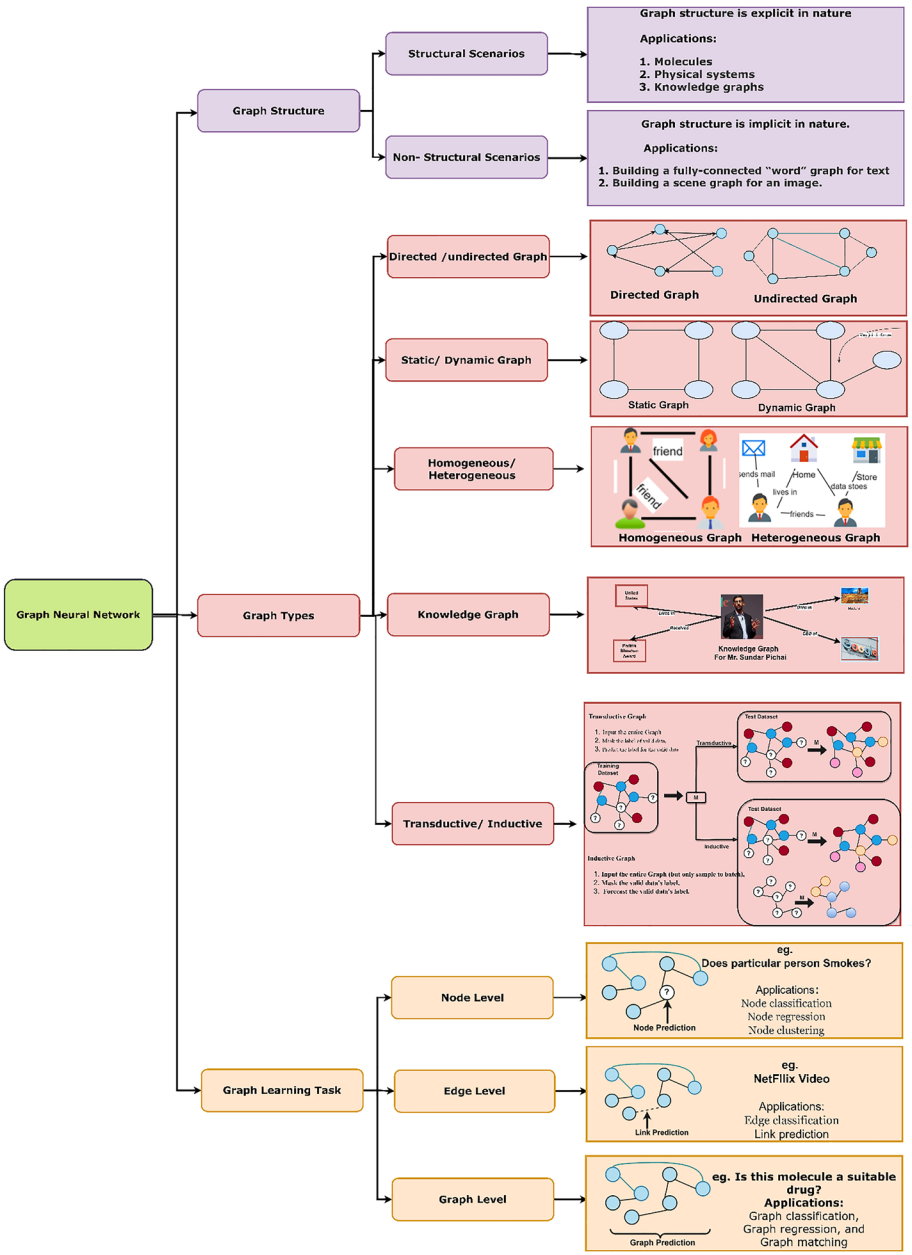

Fig. 8 Graph Neural Network Taxonomy

evolving nature of the graph. It's worth noting that different deep-learning libraries handle graph dynamics differently. TensorFlow, for instance, employs a static graph, while PyTorch utilizes a dynamic graph.

Homogeneous/heterogeneous graphs Homogeneous graphs have only one type of node and one type of edge shown in Fig. 13 (Left). A homogeneous graph is one with the same type of nodes and edges, such as an online social network with friendship as edges and nodes representing people. In homogeneous networks, nodes and edges have the same types.

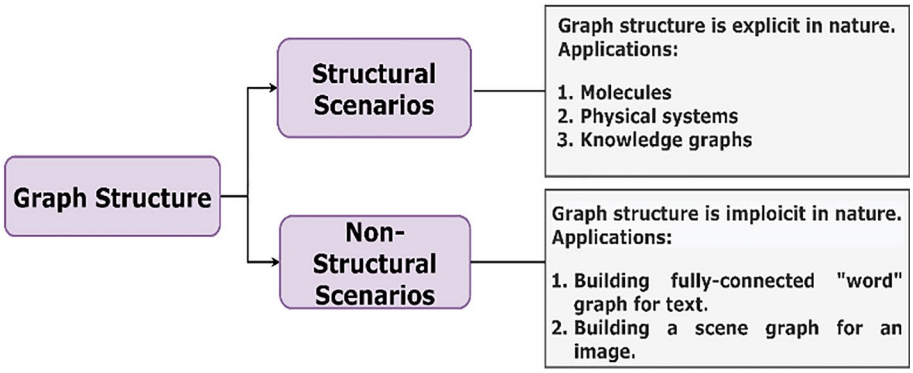

Fig. 9 Graph Structure

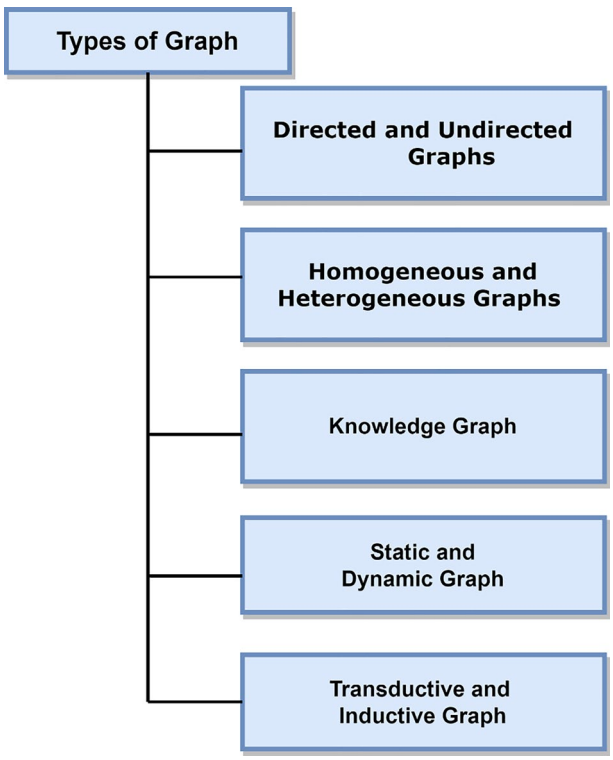

Fig. 10 Types of Graphs

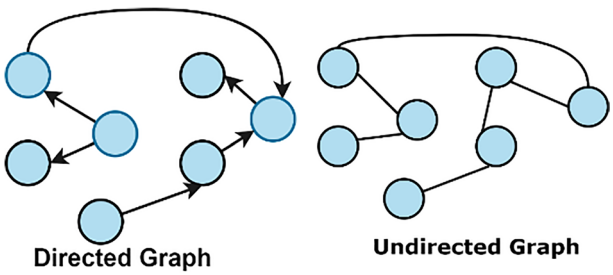

Fig. 11 Directed/Undirected Graph

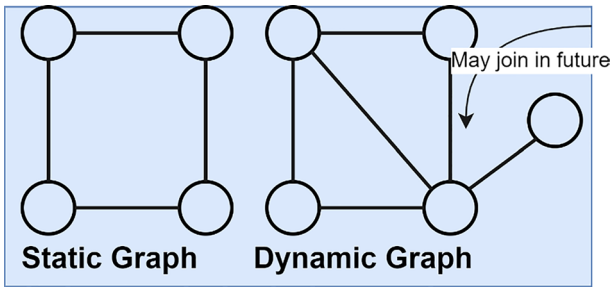

Fig. 12 Static/Dynamic Graph

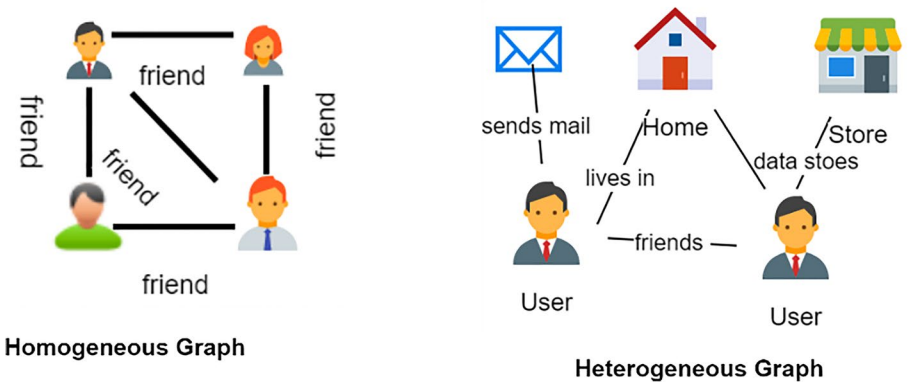

Fig. 13 Homogeneous (Left), Heterogeneous (Right) Graph

Heterogeneous graphs shown in Fig. 13 (Right) , however, have two or more different kinds of nodes and edges. A heterogeneous network is an online social network with various edges between nodes of the 'person' type, such as 'friendship' and 'co-worker.' Nodes and edges in heterogeneous graphs come in several varieties. Types of nodes and edges play critical functions in heterogeneous networks that require further consideration.

Knowledge graphs An array of triples in the form of (h, r, t) or (s, r, o) can be represented as a Knowledge Graph (KG), which is a network of entity nodes and relationship edges, with each triple (h, r, t) representing a single entity node. The relationship between an entity's head (h) and tail (t) is denoted by the r. Knowledge Graph can be considered a heterogeneous graph from this perspective. The Knowledge Graph visually depicts several real-world objects and their relationships [32]. It can be used for many new aspects, including information retrieval, knowledge-guided innovation, and answering questions [30]. Entities are objects or things that exist in the real world, including individuals, organizations, places, music tracks, movies, and people. Each relation type describes a particular relationship between various elements similarly. We can see from Fig. 14 the Knowledge graph for Mr. Sundar Pichai.

Transductive/inductive graphs In a transductive scenario shown in Fig. 15 (up), the entire graph is input, the label of the valid data is hidden, and finally, the label for the correct data is predicted. However, with an inductive graph shown in Fig. 15 (down), we

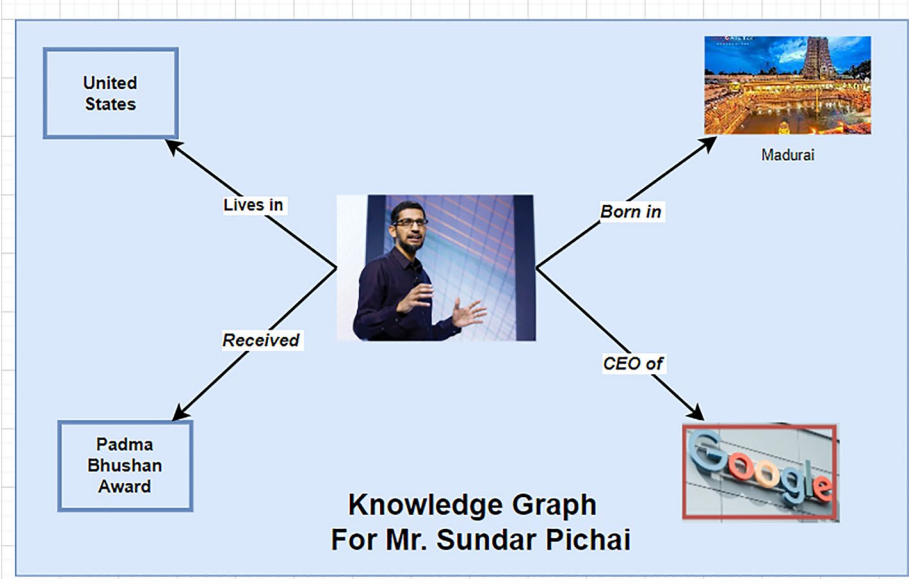

Fig. 14 Knowledge graph

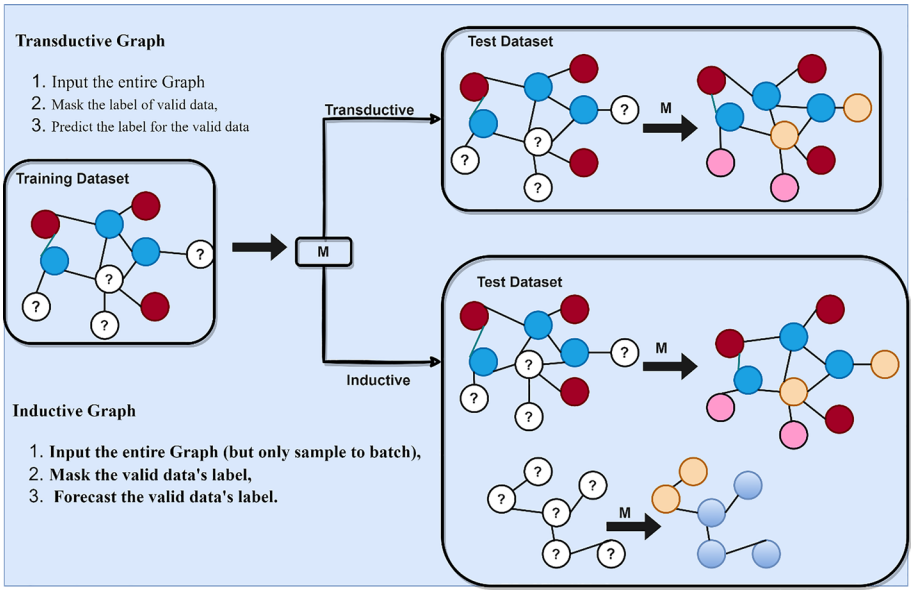

Fig. 15 Transductive/Inductive Graphs

also input the entire graph (but only sample to batch), mask the valid data's label, and forecast the valuable data's label. The model must forecast the labels of the given unlabeled nodes in a transductive context. In the inductive situation, it is possible to infer new unlabeled nodes from the same distribution.

Transductive Graph:

- In the transductive approach, the entire graph is provided as input.
- This method involves concealing the labels of the valid data.

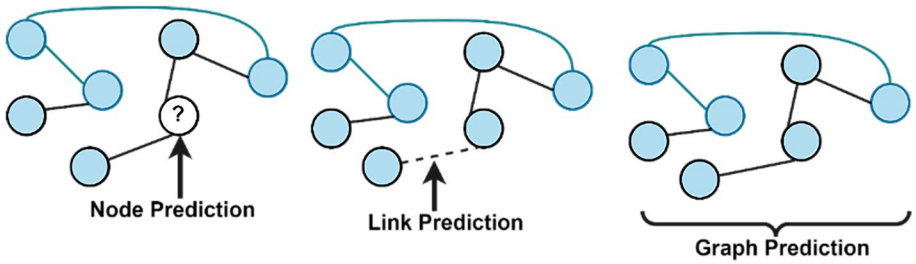

Fig. 16 Node Level Prediction (e.g., social network) (LEFT), Edge Level Prediction (e.g., Next YouTube Video?) (MIDDLE), Graph Level Prediction (e.g., molecule) (Right)

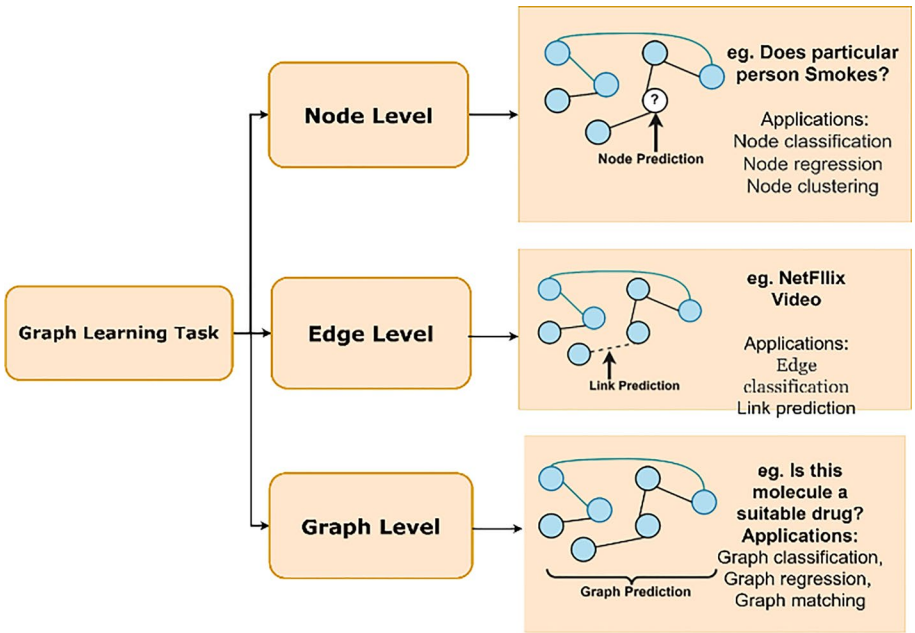

Fig. 17 Graph Learning Tasks Summary

- The primary objective is to predict the labels for the valid data.

Inductive Graph:

- The inductive approach still uses the complete graph, but only a sample within a batch is considered.
- A crucial step in this process is masking the labels of the valid data.
- The key aim here is to make predictions for the labels of the valid data.

Graph learning tasks

We perform three tasks with graphs: node classification, link prediction, and Graph Classification shown in Fig. 16.

Node-level task Node-level tasks are primarily concerned with determining the identity or function of each node within a graph. The core objective of a node-level task is to predict specific properties associated with individual nodes. For example, a node-level task in social networks could involve predicting which social group a new member is likely to join based on their connections and the characteristics of their friends' memberships. Node-level tasks are typically used when working with unlabeled data, such as identifying whether a particular individual is a smoker.

Edge-level task (link prediction) Edge-level tasks revolve around analyzing relationships between pairs of nodes in a graph. An illustrative application of an edge-level task is assessing the compatibility or likelihood of a connection between two entities, as seen in matchmaking or dating apps. Another instance of an edge-level task is evident when using platforms like Netflix, where the task involves predicting the following video to be recommended based on viewing history and user preferences.

Graph-level In graph-level tasks, the objective is to make predictions about a characteristic or property that encompasses the entire graph. For example, using a graph-based representation, one might aim to predict attributes like the olfactory quality of a molecule or its potential to bind with a disease-associated receptor. The essence of a graph-level task is to provide predictions that pertain to the graph as a whole. For instance, when assessing a newly synthesized chemical compound, a graph-level task might seek to determine whether the molecule has the potential to be an effective drug. The summary of all three learning tasks are shown in Fig. 17.

GNN models and comparative analysis of GNN models

Graph Neural Network (GNN) models represent a category of neural networks specially crafted to process data organized in graph structures. They've garnered substantial acclaim across various domains, primarily due to their exceptional capability to grasp intricate relationships and patterns within graph data. As illustrated in Fig. 18, we've outlined three distinct GNN models. A comprehensive description of these GNN models, specifically Graph Convolutional Networks (GCN), Graph Attention Networks (GAT/GAN), and GraphSAGE models can be found in the reference [33]. In Sect. "GNN models", we delve into these GNN models' intricacies; in "Comparative Study of GNN Models" section, we provide an in-depth analysis that explores their theoretical and practical aspects.

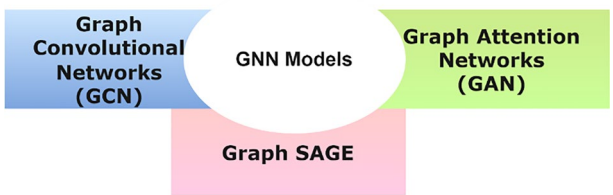

Fig. 18 GNN Models

GNN models

Graph convolution neural network (GCN)

GCN is one of the basic graph neural network variants. Thomas Kipf and Max Welling developed GCN networks. Convolution layers in Convolutional Neural Networks are essentially the same process as 'convolution' in GCNs. The input neurons are multiplied by weights called filters or kernels. The filters act as a sliding window across the image, allowing CNN to learn information from nearby cells. Weight sharing uses the same filter within the same layer throughout the image; when CNN is used to identify photos of cats vs. non-cats, the same filter is employed in the same layer to detect the cat's nose and ears. Throughout the image, the same weight (or kernel or filter in CNNs) is applied [33]. GCNs were first introduced in "Spectral Networks and Deep Locally Connected Networks on Graphs" [34].

GCNs, which learn features by analyzing neighboring nodes, carry out similar behaviors. The primary difference between CNNs and GNNs is that CNNs are made to operate on regular (Euclidean) ordered data. GNNs, on the other hand, are a generalized version of CNNs with different numbers of node connections and unordered nodes (irregular on non-Euclidean structured data). GCNs have been applied to solve many problems, for example, image classification [35], traffic forecasting [36], recommendation systems [17], scene graph generation [37], and visual question answering [38].

GCNs are particularly well-suited for tasks that involve data represented as graphs, such as social networks, citation networks, recommendation systems, and more. These networks are an extension of traditional CNNs, widely used for tasks involving grid-like data, such as images. The key idea behind GCNs is to perform convolution operations on the graph data. This enables them to capture and propagate information through the nodes in a graph by considering both a node's features and those of its neighboring nodes. GCNs typically consist of several layers, each performing convolution and aggregation steps to refine the node representations in the graph. By applying these layers iteratively, GCNs can capture complex patterns and dependencies within the graph data.

Working of graph convolutional network A Graph Convolutional Network (GCN) is a type of neural network architecture designed for processing and analyzing graph-structured data. GCNs work by aggregating and propagating information through the nodes in a graph. GCN works with the following steps shown in Fig. 19:

1) Initialization:

Each node in the graph is associated with a feature vector. Depending on the application, these feature vectors can represent various attributes or characteristics of the nodes. For example, in a social network, each node might represent a user, and the features could include user profile information.

2) Convolution Operation:

The core of a GCN is the convolution operation, which is adapted from convolutional neural networks (CNNs). It aims to aggregate information from neighboring nodes. This is done by taking a weighted sum of the feature vectors of neighboring nodes. The

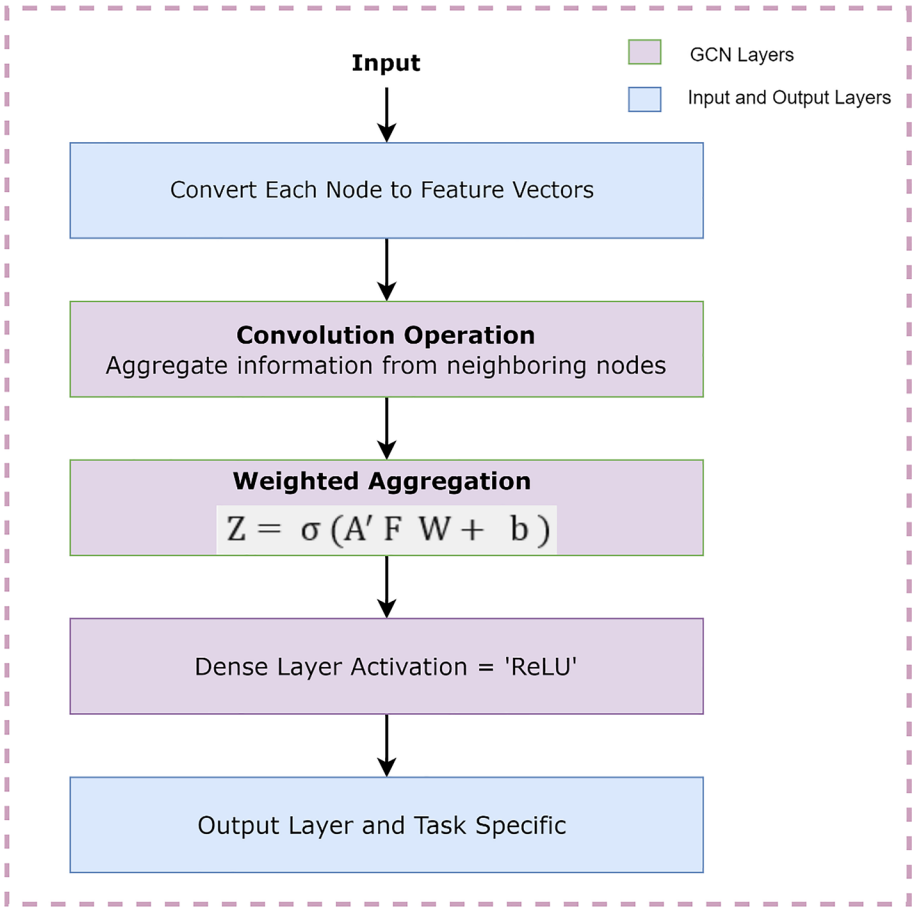

Fig. 19 Working of GCN

graph's adjacency matrix determines the weights. The resulting aggregated information is a new feature vector for each node.

3) Weighted Aggregation:

The graph's adjacency matrix, typically after normalization, provides weights for the aggregation process. In this context, for a given node, the features of its neighboring nodes are scaled by the corresponding values within the adjacency matrix, and the outcomes are then accumulated. A precise mathematical elucidation of this aggregation step is described in "Equation of GCN" section.

4) Activation function and learning weights:

The aggregated features are typically passed through an activation function (e.g., ReLU) to introduce non-linearity. The weight matrix W used in the aggregation step is learned during training. This learning process allows the GCN to adapt to the specific graph and task it is designed for.

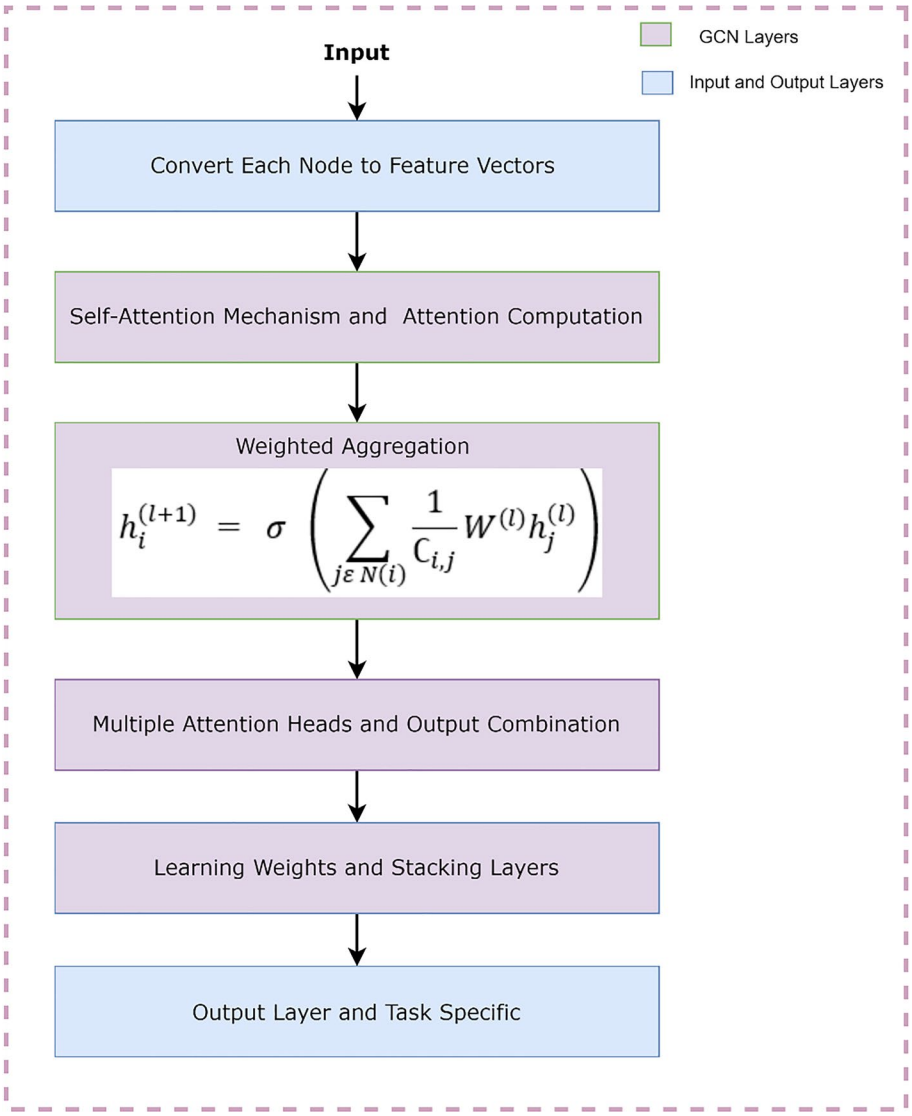

Fig. 20 Working of GAT

5) Stacking Layers:

GCNs are often used in multiple layers. This allows the network to capture more complex relationships and higher-level features in the graph. The output of one GCN layer becomes the input for the next, and this process is repeated for a predefined number of layers.

6) Task-Specific Output:

The final output of the GCN can be used for various graph-based tasks, such as node classification, link prediction, or graph classification, depending on the specific application.

Equation of GCN The Graph Convolutional Network (GCN) is based on a message-passing mechanism that can be described using mathematical equations. The core equation of a superficial, first-order GCN layer can be expressed as follows: For a graph with N nodes, let's define the following terms:

Equation 5.1 depicts a GCN layer's design. The normalized graph adjacency matrix A' and the nodes feature matrix F serve as the layer's inputs. The bias vector b and the weight matrix W are trainable parameters for the layer.

$$Z = \sigma(A^{'}FW + b) \tag{5.1}$$

When used with the design matrix, the normalized adjacency matrix effectively smoothes a node's feature vector based on the feature vectors of its close graph neighbors. This matrix captures the graph structure. A' is normalized to make each neighboring node's contribution proportional to the network's connectivity.

The layer definition is finished by applying A'FW + b to an element-wise non-linear function, such as ReLU. The downstream node classification task requires deep neural architectures to learn a complicated hierarchy of node attributes. This layer's output matrix Z can be routed into another GCN layer or any other neural network layer to do this.

Summary of graph convolution neural network (GCN) is shown in Table 2. Graph attention network (GAT/GAN)

Graph Attention Network (GAT/GAN) is a new neural network that works with graph-structured data. It uses masked self-attentional layers to address the shortcomings of past methods that depended on graph convolutions or their approximations. By stacking layers, the process makes it possible (implicitly) to assign various nodes in a neighborhood different weights, allowing nodes to focus on the characteristics of their neighborhoods without having to perform an expensive matrix operation (like inversion) or rely on prior knowledge of the graph's structure. GAT concurrently tackles numerous significant limitations of spectral-based graph neural networks, making the model suitable for both inductive and transductive applications.

Working of GAT The Graph Attention Network (GAT) is a neural network architecture designed for processing and analyzing graph-structured data shown in Fig. 20. GATs are a variation of Graph Convolutional Networks (GCNs) that incorporate the concept of attention mechanisms. GAT/GAN works with the following steps shown in Fig. 21.

1) Initialization:

As with other graph-based models, GAT starts with nodes in the graph, each associated with a feature vector. These features can represent various characteristics of the nodes.

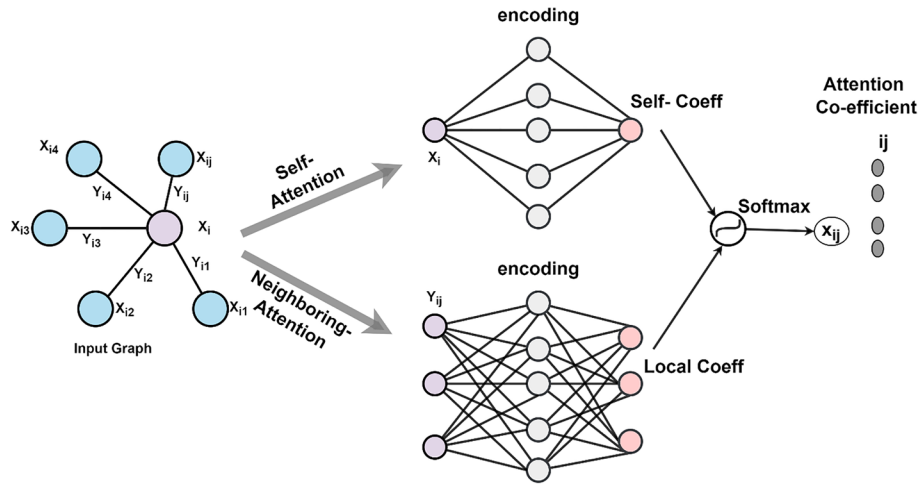

Fig. 21 How attention Coefficients updates

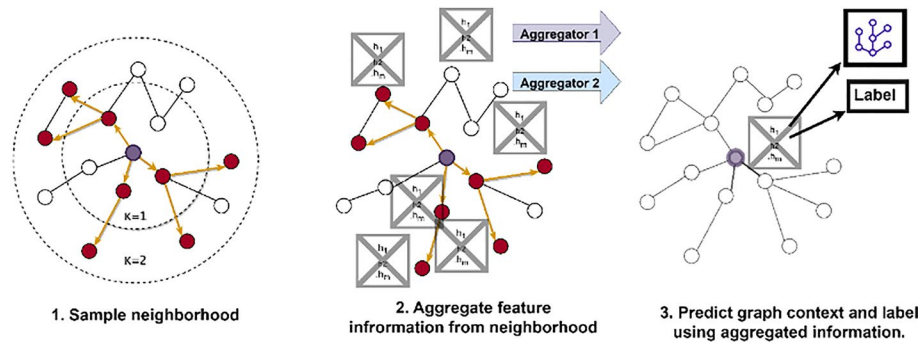

Fig. 22 Working of Graph SAGE Method

2) Self-Attention Mechanism and Attention Computation:

GAT introduces an attention mechanism similar to what is used in sequence-to-sequence models in natural language processing. The attention mechanism allows each node to focus on different neighbors when aggregating information. It assigns different attention coefficients to the neighboring nodes, making the process more flexible. For each node in the graph, GAT computes attention scores for its neighboring nodes. These attention scores are based on the features of the central node and its neighbors. The attention scores are calculated using a weighted sum of the features of the central node and its neighbors.

3) Weighted Aggregation:

The attention scores determine how much each neighbor's feature contributes to the aggregation for the central node. This weighted aggregation is carried out for all neighboring nodes, resulting in a new feature vector for the central node.

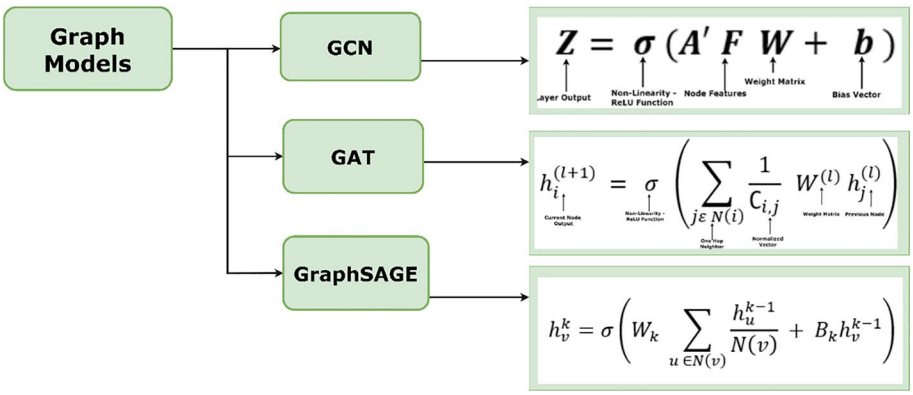

Fig. 23 Equations of GNN Models

4) Multiple Attention Heads and Output Combination:

GAT often employs multiple attention heads in parallel. Each attention head computes its attention scores and aggregation results. These multiple attention heads capture different aspects of the relationships in the graph. The outputs from the multiple attention heads are combined, typically by concatenation or averaging, to create a final feature vector for each node.

5) Learning Weights and Stacking Layers:

Similar to GCNs, GATs learn weight parameters during training. These weights are learned to optimize the attention mechanisms and adapt to the specific graph and task. GATs can be used in multiple layers to capture higher-level features and complex relationships in the graph. The output of one GAT layer becomes the input for the next layer.

The learning weights capture the importance of node relationships and contribute to information aggregation during the neighborhood aggregation process. The learning process in GNNs also relies on backpropagation and optimization algorithms. The stacking of GNN layers enables the model to capture higher-level abstractions and dependencies in the graph. Each layer refines the node representations based on information from the previous layer.

6) Task-Specific Output:

The final output of the GAT can be used for various graph-based tasks, such as node classification, link prediction, or graph classification, depending on the application.

Equation for GAT GAT's main distinctive feature is gathering data from the one-hop neighborhood [30]. A graph convolution operation in GCN produces the normalized sum of node properties of neighbors. Equation 5.2 shows the Graph attention network, which $h_i^{(l+1)}$ defines the current node output, σ denotes the non-linearity ReLU function, $j \varepsilon N(i)$ one hop neighbor, $C_{i,j}$ normalized vector, $W^{(l)}$ weight matrix, and $h_j^{(l)}$ denotes the previous node.

$$h_i^{(l+1)} = \sigma\left(\sum_{j\varepsilon N(i)} \frac{1}{C_{i,j}} W^{(l)} h_j^{(l)}\right) \tag{5.2}$$

Why is GAT better than GCN? We learned from the Graph Convolutional Network (GCN) that integrating local graph structure and node-level features results in good node classification performance. The way GCN aggregates messages, on the other hand, is structure-dependent, which may limit its use.

How attention coefficients update: the attention layer has 4 parts: [47]

1) A linear transformation: A shared linear transformation is applied to each node in the following Equation.

$$z_i^{(l)} = W^{(l)}.h_i^{(l)} \tag{5.3}$$

where h is a set of node features. W is the weight matrix. Z is the output layer node.

2) Attention Coefficients: In the GAT paradigm, it is crucial because every node can now attend to every other node, discarding any structural information. The pair-wise un-normalized attention score between two neighbors is computed in the next step. It combines the 'z' embeddings of the two nodes. Where || stands for concatenation, a learnable weight vector a(l) is put through a dot product, and a LeakyReLU is used [1]. Contrary to the dot-product attention utilized in the Transformer model, this kind of attention is called additive attention. The nodes are subsequently subjected to self-attention.

$$e_{ij}^{(l)} = \text{LeakyReLU}\left(a^{\rightarrow (l)^T}\left(z_i^{(l)} || z_j^{(l)}\right)\right) \tag{5.4}$$

3) Softmax: We utilize the softmax function to normalize the coefficients over all j values, improving their comparability across nodes.

$$\propto_{ij}^{(l)} = \frac{\exp\left(e_{ij}^{(l)}\right)}{\sum_{k\in N(i)} \exp\left(e_{ik}^{(l)}\right)} \tag{5.5}$$

4) Aggregation: This process is comparable to GCN. The neighborhood embeddings are combined and scaled based on the attention scores.

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in N(i)} \propto_{ij}^{(l)} z_j^{(l)}\right) \tag{5.6}$$

Summary of graph attention network (GAT) is shown in Table 3. GraphSAGE

GraphSAGE represents a tangible realization of an inductive learning framework shown in Fig. 22. It exclusively considers training samples linked to the training set's edges during training. This process consists of two main steps: "Sampling" and "Aggregation." Subsequently, the node representation vector is paired with the vector from the aggregated model and passed through a fully connected layer with a non-linear activation function. It's important to note that each network layer shares a standard aggregator and weight matrix. Thus, the consideration should be on the number of layers or weight matrices rather than the number of aggregators. Finally, a normalization step is applied to the layer's output.

Two major steps:

1. Sample It describes how to sample a large number of neighbors.
2. Aggregator refers to obtaining the neighbor node embedding and then determining how to collect these embeddings and change your embedding information.

Working of graphSAGE model:

1. First, initializes the eigenvectors of all nodes in the input graph
2. For each node, get its sampled neighbor nodes
3. The aggregation function is used to aggregate the information of neighbor nodes
4. And combined with embedding, Update the same by a non-linear transformation embedding Express.

Types of aggregators In the GraphSAGE method, 4 types of Aggregators are used.

1) Simple neighborhood aggregator:

$$h_v^k = \sigma\left(W_k \sum_{u \in N(v)} \frac{h_u^{k-1}}{N(v)} + B_k h_v^{k-1}\right) \tag{5.7}$$

2) Mean aggregator

$$h_v^k \leftarrow \sigma(W \cdot MEAN\left(\left\{h_i^{K-1}\right\} \cup \left\{h_u^{K-1}, \forall_u \in N(v)\right\}\right) \tag{5.8}$$

3) LSTM Aggregator: Applies LSTM to a random permutation of neighbors.
4) Pooling Aggregator: It applies a symmetric vector function and converts adjacent vectors.

$$\mathrm{AGGREGATE}_{\mathrm{k}}^{\mathrm{pool}} = \max\left(\left\{\sigma\left(\mathrm{W}_{\mathrm{pool}}\mathrm{h}_{\mathrm{ui}}^{\mathrm{k}} + \mathrm{b}\right), \forall_{\mathrm{ui}} \in \mathrm{N(v)}\right\}\right) \tag{5.9}$$

Equation of graphSAGE

$$h_v^k = \sigma\left(\left[W_k \cdot AGG\left(\left\{h_u^{k-1}, \forall_u \in N(v)\right\}\right), B_k h_v^{k-1}\right]\right) \tag{5.10}$$

Here,
$\mathrm{W_k}$, $\mathrm{B_k}$: is learnable weight matrices.
$W_k B_k$ = is learnable wight matrices.
$h_v^0 = x_v$: $initial0-$ the layer embeddings are equal to node features.
h_u^{k-1} = Generalized Aggregation.
$z_v = h_v^k n$: embedding after k layers of neighborhood aggregation.
$\sigma-$ non linearity (ReLU).

Summary of graphSAGE is shown in Table 4. Comparative study of GNN models

Comparison based on practical implementation of GNN models

Table 5 describes the dataset statistics for different datasets used in literature for graph type of input. The datasets are CORA, Citeseer, and Pubmed. These statistics provide information about the kind of dataset, the number of nodes and edges, the number of classes, the number of features, and the label rate for each dataset. These details are essential for understanding the characteristics and scale of the datasets used in the context of citation networks. Comparison of the GNN model with equation in shown in Fig. 23.

Table 6 shows the performance results of different Graph Neural Network (GNN) models on various datasets. Table 6 provides accuracy scores for other GNN models on different datasets. Additionally, the time taken for some models to compute results is indicated in seconds. This information is crucial for evaluating the performance of these models on specific datasets.

Table 5 Different Dataset Statistics of Citation Network [33]

Dataset Statistics			
Datasets	**CORA**	**Citeseer**	**Pubmed**
Type	Citation network	Citation network	Citation network
Nodes	2708	3327	19717
Edges	5429	4732	44338
Classes	7	6	3
Features	1433	3703	500
Label rate	0.052	0.036	0.003

Table 6 Performance metrics of different models with different datasets [33, 47, 48]

Dataset	Model					
	GCN	GAT	GraphSAGE			
			GraphSAGE-Simple	GraphSAGE—Mean	GraphSAGE—LSTM	GraphSAGE—Pooling
CORA	81.5 (4 s)	83	76.8	78.7	79.7	80.7
Citeseer	70.3 (7 s)	72.5	74.2	77.8	78.8	79.8
Pubmed	79.0 (38 s)	79				

Comparison based on theoretical concepts of GNN models are described in Table 7.

Graph neural network applications

Graph construction

Graph Neural Networks (GNNs) have a wide range of applications spanning diverse domains, which encompass modern recommender systems, computer vision, natural language processing, program analysis, software mining, bioinformatics, anomaly detection, and urban intelligence, among others. The fundamental prerequisite for GNN utilization is the transformation or representation of input data into a graph-like structure. In the realm of graph representation learning, GNNs excel in acquiring essential node or graph embeddings that serve as a crucial foundation for subsequent tasks [61].

The construction of a graph involves a two-fold process:

1) Graph creation and
2) Learning about graph representations
3) Graph Creation: The generation of graphs is essential for depicting the intricate relationships embedded within diverse incoming data. With the varied nature of input data, various applications adopt techniques to create meaningful graphs. This process is indispensable for effectively communicating the structural nuances of the data, ensuring the nodes and edges convey their semantic significance, particularly tailored to the specific task at hand.
4) Learning about graph representations: The subsequent phase involves utilizing the graph expression acquired from the input data. In GNN-based Learning for graph representations, some studies employ well-established GNN models like GraphSAGE, GCN, GAT, and GGNN, which offer versatility for various application tasks. However, when faced with specific tasks, it may be necessary to customize the GNN architecture to address particular challenges more effectively.

The different application which is considered a graph

1) Molecular Graphs: Atoms and electrons serve as the basic building blocks of matter and molecules, organized in three-dimensional structures. While all particles interact, we primarily acknowledge a covalent connection between two stable atoms when they are sufficiently spaced apart. Various atom-to-atom bond configurations exist, including single and double bonds. This three-dimensional arrangement is con-

Table 7 Comparison of the GNN model with Advantages, Disadvantages, and application areas: [30]

Model	Features/ characteristics	Message Passing Mechanism	Attention Mechanism	Aggregation Strategy	Scalability	Use Cases	Advantages	Disadvantages
Graph Convolution Neural Network (GCN)	• Propagates information through neighbors • The simple and basic method • It uses a linear transformation • Homophily (focuses on immediate neighbors) • Smoothness assumption (neighbors should have similar representations)	Fixed weighted sum	No attention	Aggregation shown in	Limited	• Node classification • Semi-supervised Learning • Recommendation systems	• Simplicity & interpretability • Stable training often requires fewer epochs	• We have limited expressive power • Inability to capture local structures • No edge features
Graph Attention Network (GAT)	• Learns weights for each neighbor's message • Used for transductive and inductive Learning, i.e., you can work with graph structures you've never seen before • Can handle varying neighborhood sizes	Weighted sum with learned weights	Self-attention	Aggregation	Moderate	• Node classification • Link prediction • Any task requiring localized information	• Ability to capture fine-grained relationships • Improved performance on tasks requiring attention to specific neighbors	• Computationally more expensive than GCN • It can be more sensitive to hyperparameters

Table 7 (continued)

Model	Features/ characteristics	Message Passing Mechanism	Attention Mechanism	Aggregation Strategy	Scalability	Use Cases	Advantages	Disadvantages
GraphSAGE	• Aggregates information from sampled neighbors • The number of model parameters is independent of the number of graph nodes. This makes GraphSAGE able to handle larger graphs • It can handle both supervised and unsupervised tasks • Sample-based approach with random or predefined sampling strategies	Fixed weighted sum	No attention	Aggregation	Highly scalable	• Node classification • Link prediction • Tasks where scalability is crucial	• Scalability to large graphs • Flexible sampling strategies • Suitable for graphs with varying node degrees	• Limited ability to capture global graph structures • May require more epochs to train effectively

veniently and commonly represented as a graph, with atoms representing nodes and covalent bonds representing edges [62].

2) Graphs of social networks: These networks are helpful research tools for identifying trends in the collective behavior of individuals, groups, and organizations. We may create a graph that represents groupings of people by visualizing individuals as nodes and their connections as edges [63].
3) Citation networks as graphs: When they publish papers, scientists regularly reference the work of other scientists. Each manuscript can be visualized as a node in a graph of these citation networks, with each directed edge denoting a citation from one publication to another. Additionally, we can include details about each document in each node, such as an abstract's word embedding [64].
4) Within computer vision: We may want to tag certain things in visual scenes. Then, we can construct graphs by treating these things as nodes and their connections as edges.

GNNs are used to model data as graphs, allowing for the capture of complex relationships and dependencies that traditional machine learning models may struggle to represent. This makes GNNs a valuable tool for tasks where data has an inherent graph structure or where modeling relationships is crucial for accurate predictions and analysis.

Graph neural networks (GNNs) applications in different fields

NLP (natural language processing)

a) Document Classification: GNNs can be used to model the relationships between words or sentences in documents, allowing for improved document classification and information retrieval.
b) Text Generation: GNNs can assist in generating coherent and contextually relevant text by capturing dependencies between words or phrases.
c) Question Answering: GNNs can help in question-answering tasks by representing the relationships between question words and candidate answers within a knowledge graph.
d) Sentiment Analysis: GNNs can capture contextual information and sentiment dependencies in text, improving sentiment analysis tasks.

Computer vision

a) Image Segmentation: GNNs can be employed for pixel-level image segmentation tasks by modeling relationships between adjacent pixels as a graph.
b) Object Detection: GNNs can assist in object detection by capturing contextual information and relationships between objects in images.
c) Scene Understanding: GNNs are used for understanding complex scenes and modeling spatial relationships between objects in an image.

Bioinformatics

a) Protein-Protein Interaction Prediction: GNNs can be applied to predict interactions between proteins in biological networks, aiding in drug discovery and understanding disease mechanisms.
b) Genomic Sequence Analysis: GNNs can model relationships between genes or genetic sequences, helping in gene expression prediction and sequence classification tasks.
c) Drug Discovery: GNNs can be used for drug-target interaction prediction and molecular property prediction, which is vital in pharmaceutical research.

Table 8 offers a concise overview of various research papers that utilize Graph Neural Networks (GNNs) in diverse domains, showcasing the applications and contributions of GNNs in each study.

Table 8 Different application areas with their proposed methodology of Graph Neural Networks

Ref	Application Area	Proposed Methodology	GNN Model applied
[58] (2022), [59] (2020)	Recommender Systems	User/item representations Recommendation system based on heterogeneous features	GCN, GAT, GraphSAGE
[67–69] (2021)	Natural Language Processing	Text graph transformer for document classification Text-Based Relational Reasoning Semantic parsing	graph2seq, graph2tree, graph2graph
[63], (2021) [64] (2022)	HealthCare	Data Analysis-Based Agricultural Products Management Immunization and vaccine injury	GCN
[72] (2017) [73] (2020)	Natural Language Processing	Knowledge Base Completion of text Knowledge Graph Alignment of text	GNN-LSTM
[67], (2021) [68] (2020)	Computer Vision	Image and video understanding 3D object detection in a point cloud	GCN, GAT
[76] (2021)	Anomaly Detection	Industrial Internet of Things	GCN, GAN, and GraphSAGE
[29] (2023)	Traffic Forecasting	Hybrid GCN and branch-and-bound optimization for traffic flow forecasting	GCN
[77] (2023)	HealthCare	The configuration of fMRI-derived networks determines the effectiveness of a graph neural network in discerning patients with major depressive disorder through classification	GNN (Text based)
[28] (2023)	Traffic Prediction	A study focusing on the prediction of multi-port ship traffic through the application of Spatiotemporal Graph Neural Networks	GNN

Table 9 highlights various applications of GNNs in Natural Language Processing, Computer Vision, and Bioinformatics domains, showcasing how GNN models are adapted and used for specific tasks within each field.

Future directions of graph neural network

The contribution of the existing literature to GNN principles, models, datasets, applications, etc., was the main emphasis of this survey. In this section, several potential future study directions are suggested. Significant challenges have been noted, including unbalanced datasets, the effectiveness of current methods, text classification, etc. We have also looked at the remedies to address these problems. We have suggested future and advanced directions to address these difficulties regarding domain adaptation, data augmentation, and improved classification. Table 10 displays future directions.

1) Imbalanced Datasets—Limited labeled data, domain-dependent data, and imbalanced data are currently issues with available datasets. Transfer learning and domain adaptation are solutions to these issues.
2) Accuracy of Existing Systems/Models—can utilize deep learning models such as GCN, GAT, and GraphSAGE approaches to increase the efficiency and precision of current systems. Additionally, training models on sizable, domain-specific datasets can enhance performance.
3) Enhancing Text Classification: Text classification poses another significant challenge, which is effectively addressed by leveraging advanced deep learning methodologies like graph neural networks, contributing to the improvement of text classification accuracy and performance.

The above Table 10 describes the research gaps and future directions presented in the above literature. These research gaps and future directions highlight the challenges and proposed solutions in the field of text classification and structural analysis.

Table 11 provides an overview of different research papers, their publication years, the applications they address, the graph structures they use, the graph types, the graph tasks, and the specific Graph Neural Network (GNN) models utilized in each study.

Conclusions

Graph Neural Networks (GNNs) have witnessed rapid advancements in addressing the unique challenges presented by data structured as graphs, a domain where conventional deep learning techniques, originally designed for images and text, often struggle to provide meaningful insights. GNNs offer a powerful and intuitive approach that finds broad utility in applications relying on graph structures. This comprehensive survey on GNNs offers an in-depth analysis covering critical aspects such as GNN fundamentals, the interplay with convolutional neural networks, GNN message-passing mechanisms, diverse GNN models, practical use cases, and a forward-looking perspective. Our central focus is on elucidating the foundational characteristics of GNNs, a field teeming with contemporary applications that continually enhance our comprehension and utilization of this technology.

Table 9 Different Domains with their Tasks in Graph Neural Networks

Refs.	Technology domain	Task	Details	GNN Model applied
[78] (2021)	Natural Language Processing	Text Sentiment Analysis	Their innovation involved introducing a multi-level graph neural network (MLGNN) tailored for text sentiment analysis. Their approach effectively incorporated both local and global features, utilizing node connection windows of varying sizes across different levels. Additionally, they seamlessly integrated a scaled dot-product attention mechanism as a means of message passing within their method, allowing for the integration of features from individual word nodes in the graph	MLGNN GAT
[79] (2022)	Natural Language Processing	Text Classification	GNNs were chosen for their aptness in handling 2D vectors, which aligns with the two-dimensional nature of text data. In their approach, Self-Organizing Maps (SOM) was employed to determine the closest neighbors within the graphs, facilitating the computation of actual distances between these neighboring elements	GNN Self-Organizing Maps (For calculating distance)
[80] (2023)	Natural Language Processing	Question Generation	They created a graph from the input text, where nodes represent words or phrases, and edges show their relationships. An auto-encoder model compresses the graph, capturing key information. This compressed representation helps generate context-relevant questions by selecting nodes and edges dynamically	Context-Aware Auto-Encoded Graph Neural Model

Table 9 (continued)

Refs.	Technology domain	Task	Details	GNN Model applied
[81]–[83] (2022)	Computer Vision	Graph Construction	There are three methods for graph construction 1. Segmenting the image or video frame into uniform grid sections, with each grid section serving as an individual vertex within the visual graph 2. Utilizing preprocessed structures, like scene graphs, for direct vertex representation 3. Incorporating semantic information to group visually similar pixel features into the same vertex	GNN
[67], (2021) [68] (2020)	Computer Vision	3D object detection	Image and video understanding. 3D object detection in a point cloud	GCN, GAT
[84] (2021)	Bioinformatics	Multispecies Protein Function Prediction	DeepGraph Go has 3 Features: 1. InterPro for representation vector 2. Multiple graph convolutional neural (GCN) layers 3. Multispecies strategy	DeepGraphGO: A semi-supervised deep learning approach that harnesses the strengths of both protein sequence and network data by utilizing a graph neural network (GNN)
[85] (2022)	Bioinformatics	Link Prediction in Biomedical Networks	1. Leveraging GCN to extract node-specific features from both sequence and structural data 2. Employing a GCN-based encoder to enhance the node features by capturing inter-node dependencies within the network effectively 3. Pre-training the node features using graph reconstruction tasks as a foundational step	Pre-Training Graph Neural Networks- (PT-GNN)

Table 9 (continued)

Refs.	Technology domain	Task	Details	GNN Model applied
[86] (2022)	Bioinformatics	Predicting Drug–Protein Interactions	The network undergoes optimization through supervised signals derived from the downstream task, specifically the DPI prediction. By engaging in information propagation within the drug-protein association network, a Graph Neural Network can grasp network-level insights encompassing a variety of drugs and proteins. This approach amalgamates network-level information with learning-based techniques	Bridge Drug–Protein Interactions (Bridge-DPI)
[87] (2022)	Bioinformatics	Predicting Molecular	LR-GNN utilizes a graph convolutional network (GCN) encoder to acquire node embeddings. To depict the relationships between molecules, a propagation rule has been crafted to encapsulate the node embeddings at each GCN-encoder layer, forming the LR representation	Link Representation (LR-GNN)

Table 10 A list of research gaps and future research directions

Sr. no.	Research gaps	Future directions
1	Lack of ready datasets Inconsistent Datasets	Domain Adaptation
2	Inefficient and time-consuming feature extraction task Improving Text Classification	Here combining, deep learning and machine learning methods like GNNs to increase classification accuracy
3	Accuracy of Existing Systems/ Models Identification of Type of structure, i.e., homogenous heterogeneous	Deep Learning models such as GCN, GAT, and GraphSAGE

The continuous evolution of GNN-based research has underscored the growing need to address issues related to graph analysis, which we aptly refer to as the frontiers of GNNs. In our exploration, we delve into several crucial recent research domains within the realm of GNNs, encompassing areas like link prediction, graph generation, and graph categorization, among others.

Table 11 Summary of Graph Neural Networks with application area, graph structure, type, task, and model used

Refs.	Application	Graph structure	Graph type	Graph task	GNN model used
[88] (2020)	1. Recurrent Graph Neural Networks for Text Classification	Structural data	Static Graph	Node Classification	Text GCN RGNN
[68] (2021)	1. Machine translation 2. Natural language generation 3. Information extraction 4. Semantic parsing	Structural data	Static Graph	Node & Edge Level task	Graph2seq Graph2tree Graph2graph
[16] (2019)	1. Multi-hop Reading Comprehension	Structural data	Heterogeneous Graphs	Edge Level task	GCN
[89] (2020)	1. Edge masking	Structural data	Undirected Graph	Edge Level task	LSTM + GNN
[90] (2020)	1. Multi-hop reading comprehension on hotpot a Fact verification on FEVER	Structural data	Directed Graph	Node level task	GCN

Appendix

See Tables 12 and 13

Table 12 Commonly Used Datasets in this Survey (Related to Graph)

Application Area	Datasets	Refs.
Citation Networks	1) Pubmed 2) Cora 3) Citeseer 4) NELL	[22, 31, 47, 48]
Social Networks	1) Reddit 2) Ciao 3) Epinions 4) Microblogs	[17, 31, 52, 57, 59]

Table 13 Python Libraries for Graph Computing

Sr. No	Python library	GitHub Link
1	PyTorch Geometric	https://github.com/pyg-team/pytorch_geometric
2	Deep Graph Library	https://www.dgl.ai
3	GraphVite	https://graphvite.io
4	Plato	https://github.com/baskerville/plato
5	Paddle graph learning	https://github.com/PaddlePaddle/PGL

Abbreviations

GNN	Graph Neural Network
GCN	Graph Convolution Network
GAT	Graph Attention Networks
NLP	Natural Language Processing
GNN	Graph Neural Network
CNN	Convolution Neural Networks
RNN	Recurrent Neural Networks
ML	Machine Learning
DL	Deep Learning
KG	Knowledge Graph

Acknowledgements
I am grateful to all of those with whom I have had the pleasure to work during this research work. Each member has provided me extensive personal and professional guidance and taught me a great deal about scientific research and life in general.

Author contributions
Conceptualization, BK and SP; methodology, BK and SP; software, BK; validation, BK, SP, KK; formal analysis, BK; investigation, BK; resources, BK; data curation, BK and SP; writing—original draft preparation, BK; writing—review and editing, SP, KK, and ST; visualization, BK; supervision, SP; project administration, SP, ST; funding acquisition, KK.

Funding
This work was supported by the Research Support Fund (RSF) of Symbiosis International (Deemed University), Pune, India.

Availability of data and materials
Not applicable.

Declarations

Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Competing interests
The authors declare that they have no competing interests**.**

Received: 28 June 2023 Accepted: 27 December 2023
Published online: 16 January 2024

References

1. Pucci A, Gori M, Hagenbuchner M, Scarselli F, Tsoi AC. Investigation into the application of graph neural networks to large-scale recommender systems, infona.pl, no. 32, no 4, pp. 17–26, 2006.
2. Mahmud FB, Rayhan MM, Shuvo MH, Sadia I, Morol MK. A comparative analysis of Graph Neural Networks and commonly used machine learning algorithms on fake news detection, Proc. - 2022 7th Int. Conf. Data Sci. Mach. Learn. Appl. CDMA 2022, pp. 97–102, 2022.
3. Cui L, Seo H, Tabar M, Ma F, Wang S, Lee D, Deterrent: Knowledge Guided Graph Attention Network for Detecting Healthcare Misinformation, Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 492–502, 2020.
4. Gori M, Monfardini G, Scarselli F, A new model for earning in raph domains, Proc. Int. Jt. Conf. Neural Networks, vol. 2, no. January 2005, pp. 729–734, 2005, https://doi.org/10.1109/IJCNN.2005.1555942.
5. Scarselli F, Yong SL, Gori M, Hagenbuchner M, Tsoi AC, Maggini M. Graph neural networks for ranking web pages, Proc.—2005 IEEE/WIC/ACM Int. Web Intell. WI 2005, vol. 2005, no. January, pp. 666–672, 2005, doi: https://doi.org/10.1109/WI.2005.67.
6. Gandhi S, Zyer AP, P3: Distributed deep graph learning at scale, Proc. 15th USENIX Symp. Oper. Syst. Des. Implementation, OSDI 2021, pp. 551–568, 2021.
7. Li C, Guo J, Zhang H. Pruning neighborhood graph for geodesic distance based semi-supervised classification, in 2007 International Conference on Computational Intelligence and Security (CIS 2007), 2007, pp. 428–432.
8. Zhang Z, Cui P, Pei J, Wang X, Zhu W, Eigen-gnn: A graph structure preserving plug-in for gnns, IEEE Trans. Knowl. Data Eng., 2021.
9. Nandedkar AV, Biswas PK. A granular reflex fuzzy min–max neural network for classification. IEEE Trans Neural Netw. 2009;20(7):1117–34.
10. Chaturvedi DK, Premdayal SA, Chandiok A. Short-term load forecasting using soft computing techniques. Int'l J Commun Netw Syst Sci. 2010;3(03):273.
11. Hashem T, Kulik L, Zhang R. Privacy preserving group nearest neighbor queries, in Proceedings of the 13th International Conference on Extending Database Technology, 2010, pp. 489–500.

12. Sun Z et al. Knowledge graph alignment network with gated multi-hop neighborhood aggregation, in Proceedings of the AAAI Conference on Artificial Intelligence, 2020, vol. 34, no. 01, pp. 222–229.
13. Zhang M, Chen Y. Link prediction based on graph neural networks. Adv Neural Inf Process Syst. 31, 2018.
14. Stanimirović PS, Katsikis VN, Li S. Hybrid GNN-ZNN models for solving linear matrix equations. Neurocomputing. 2018;316:124–34.
15. Stanimirović PS, Petković MD. Gradient neural dynamics for solving matrix equations and their applications. Neurocomputing. 2018;306:200–12.
16. Zhang C, Song D, Huang C, Swami A, Chawla NV. Heterogeneous graph neural network, in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 793–803.
17. Fan W et al. Graph neural networks for social recommendation," in The world wide web conference, 2019, pp. 417–426.
18. Gui T et al. A lexicon-based graph neural network for Chinese NER," in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 1040–1050.
19. Qasim SR, Mahmood H, Shafait F. Rethinking table recognition using graph neural networks, in 2019 International Conference on Document Analysis and Recognition (ICDAR), 2019, pp. 142–147
20. You J, Ying R, Leskovec J. Position-aware graph neural networks, in International conference on machine learning, 2019, pp. 7134–7143.
21. Cao D, et al. Spectral temporal graph neural network for multivariate time-series forecasting. Adv Neural Inf Process Syst. 2020;33:17766–78.
22. Xhonneux LP, Qu M, Tang J. Continuous graph neural networks. In International Conference on Machine Learning, 2020, pp. 10432–10441.
23. Zhou K, Huang X, Li Y, Zha D, Chen R, Hu X. Towards deeper graph neural networks with differentiable group normalization. Adv Neural Inf Process Syst. 2020;33:4917–28.
24. Gu F, Chang H, Zhu W, Sojoudi S, El Ghaoui L. Implicit graph neural networks. Adv Neural Inf Process Syst. 2020;33:11984–95.
25. Liu Y, Guan R, Giunchiglia F, Liang Y, Feng X. Deep attention diffusion graph neural networks for text classification. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, 2021, pp. 8142–8152.
26. Gasteiger J, Becker F, Günnemann S. Gemnet: universal directional graph neural networks for molecules. Adv Neural Inf Process Syst. 2021;34:6790–802.
27. Yao D et al. Deep hybrid: multi-graph neural network collaboration for hyperspectral image classification. Def. Technol. 2022.
28. Li Y, et al. Research on multi-port ship traffic prediction method based on spatiotemporal graph neural networks. J Mar Sci Eng. 2023;11(7):1379.
29. Djenouri Y, Belhadi A, Srivastava G, Lin JC-W. Hybrid graph convolution neural network and branch-and-bound optimization for traffic flow forecasting. Futur Gener Comput Syst. 2023;139:100–8.
30. Zhou J, et al. Graph neural networks: a review of methods and applications. AI Open. 2020;1(January):57–81. https://doi.org/10.1016/j.aiopen.2021.01.001.
31. Rong Y, Huang W, Xu T, Huang J. Dropedge: Towards deep graph convolutional networks on node classification. arXiv preprint arXiv:1907.10903. 2019.
32. Abu-Salih B, Al-Qurishi M, Alweshah M, Al-Smadi M, Alfayez R, Saadeh H. Healthcare knowledge graph construction: a systematic review of the state-of-the-art, open issues, and opportunities. J Big Data. 2023;10(1):81.
33. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. arXiv Prepr. arXiv1609.02907, 2016.
34. Berg RV, Kipf TN, Welling M. Graph Convolutional Matrix Completion. 2017, http://arxiv.org/abs/1706.02263
35. Monti F, Boscaini D, Masci J, Rodola E, Svoboda J, Bronstein MM. Geometric deep learning on graphs and manifolds using mixture model cnns. InProceedings of the IEEE conference on computer vision and pattern recognition 2017 (pp. 5115-5124).
36. Cui Z, Henrickson K, Ke R, Wang Y. Traffic graph convolutional recurrent neural network: a deep learning framework for network-scale traffic learning and forecasting. IEEE Trans Intell Transp Syst. 2020;21(11):4883–94. https://doi.org/10.1109/TITS.2019.2950416.
37. Yang J, Lu J, Lee S, Batra D, Parikh D. Graph r-cnn for scene graph generation. InProceedings of the European conference on computer vision (ECCV) 2018 (pp. 670-685). https://doi.org/10.1007/978-3-030-01246-5_41.
38. Teney D, Liu L, van Den Hengel A. Graph-structured representations for visual question answering. InProceedings of the IEEE conference on computer vision and pattern recognition 2017 (pp. 1-9). https://doi.org/10.1109/CVPR.2017.344.
39. Yao L, Mao C, Luo Y. Graph convolutional networks for text classification. Proc AAAI Conf Artif Intell. 2019;33(01):7370–7.
40. De Cao N, Aziz W, Titov I. Question answering by reasoning across documents with graph convolutional networks. arXiv Prepr. arXiv1808.09920, 2018.
41. Gao H, Wang Z, Ji S. Large-scale learnable graph convolutional networks. in Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 1416–1424.
42. Hu F, Zhu Y, Wu S, Wang L, Tan T. Hierarchical graph convolutional networks for semi-supervised node classification. arXiv Prepr. arXiv1902.06667, 2019.
43. Lange O, Perez L. Traffic prediction with advanced graph neural networks. DeepMind Research Blog Post, https://deepmind.google/discover/blog/traffic-prediction-with-advanced-graph-neural-networks/. 2020.
44. Duan C, Hu B, Liu W, Song J. Motion capture for sporting events based on graph convolutional neural networks and single target pose estimation algorithms. Appl Sci. 2023;13(13):7611.
45. Balcıoğlu YS, Sezen B, Çerasi CC, Huang SH. machine design automation model for metal production defect recognition with deep graph convolutional neural network. Electronics. 2023;12(4):825.

46. Baghbani A, Bouguila N, Patterson Z. Short-term passenger flow prediction using a bus network graph convolutional long short-term memory neural network model. Transp Res Rec. 2023;2677(2):1331–40.
47. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y. Graph attention networks. Stat. 2017;1050(20):10–48550.
48. Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. Advances in neural information processing systems. 2017; 30.
49. Ye Y, Ji S. Sparse graph attention networks. IEEE Trans Knowl Data Eng. 2021;35(1):905–16.
50. Chen Z et al. Graph neural network-based fault diagnosis: a review. arXiv Prepr. arXiv2111.08185, 2021.
51. Brody S, Alon U, Yahav E. How attentive are graph attention networks? arXiv Prepr. arXiv2105.14491, 2021.
52. Huang J, Shen H, Hou L, Cheng X. Signed graph attention networks," in International Conference on Artificial Neural Networks. 2019, pp. 566–577.
53. Seraj E, Wang Z, Paleja R, Sklar M, Patel A, Gombolay M. Heterogeneous graph attention networks for learning diverse communication. arXiv preprint arXiv: 2108.09568. 2021.
54. Zhang Y, Wang X, Shi C, Jiang X, Ye Y. Hyperbolic graph attention network. IEEE Transactions on Big Data. 2021;8(6):1690–701.
55. Yang X, Ma H, Wang M. Research on rumor detection based on a graph attention network with temporal features. Int J Data Warehous Min. 2023;19(2):1–17.
56. Lan W, et al. KGANCDA: predicting circRNA-disease associations based on knowledge graph attention network. Brief Bioinform. 2022;23(1):bbab494.
57. Xiao L, Wu X, Wang G, 2019, December. Social network analysis based on graph SAGE. In 2019 12th international symposium on computational intelligence and design (ISCID) (Vol. 2, pp. 196–199). IEEE.
58. Chang L, Branco P. Graph-based solutions with residuals for intrusion detection: The modified e-graphsage and e-resgat algorithms. arXiv preprint arXiv:2111.13597. 2021.
59. Oh J, Cho K, Bruna J. Advancing graphsage with a data-driven node sampling. arXiv preprint arXiv:1904.12935. 2019.
60. Kapoor M, Patra S, Subudhi BN, Jakhetiya V, Bansal A. Underwater Moving Object Detection Using an End-to-End Encoder-Decoder Architecture and GraphSage With Aggregator and Refactoring. InProceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2023 (pp. 5635-5644).
61. Bhatti UA, Tang H, Wu G, Marjan S, Hussain A. Deep learning with graph convolutional networks: An overview and latest applications in computational intelligence. Int J Intell Syst. 2023;2023:1–28.
62. David L, Thakkar A, Mercado R, Engkvist O. Molecular representations in AI-driven drug discovery: a review and practical guide. J Cheminform. 2020;12(1):1–22.
63. Davies A, Ajmeri N. Realistic Synthetic Social Networks with Graph Neural Networks. arXiv preprint arXiv:2212.07843. 2022; 15.
64. Frank MR, Wang D, Cebrian M, Rahwan I. The evolution of citation graphs in artificial intelligence research. Nat Mach Intell. 2019;1(2):79–85.
65. Gao C, Wang X, He X, Li Y. Graph neural networks for recommender system. InProceedings of the Fifteenth ACM International Conference on Web Search and Data Mining 2022 (pp. 1623-1625).
66. Wu S, Sun F, Zhang W, Xie X, Cui B. Graph neural networks in recommender systems: a survey. ACM Comput Surv. 2022;55(5):1–37.
67. Wu L, Chen Y, Shen K, Guo X, Gao H, Li S, Pei J, Long B. Graph neural networks for natural language processing: a survey. Found Trends Mach Learn. 2023;16(2):119–328.
68. Wu L, Chen Y, Ji H, Liu B. Deep learning on graphs for natural language processing. InProceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval 2021 (pp. 2651-2653).
69. Liu X, Su Y, Xu B. The application of graph neural network in natural language processing and computer vision. In2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI) 2021 (pp. 708-714).
70. Harmon SHE, Faour DE, MacDonald NE. Mandatory immunization and vaccine injury support programs: a survey of 28 GNN countries. Vaccine. 2021;39(49):7153–7.
71. Yan W, Zhang Z, Zhang Q, Zhang G, Hua Q, Li Q. Deep data analysis-based agricultural products management for smart public healthcare. Front Public Health. 2022;10:847252.
72. Hamaguchi T, Oiwa H, Shimbo M, Matsumoto Y. Knowledge transfer for out-of-knowledge-base entities: a graph neural network approach. arXiv preprint arXiv:1706.05674. 2017.
73. Dai D, Zheng H, Luo F, Yang P, Chang B, Sui Z. Inductively representing out-of-knowledge-graph entities by optimal estimation under translational assumptions. arXiv preprint arXiv:2009.12765.
74. Pradhyumna P, Shreya GP. Graph neural network (GNN) in image and video understanding using deep learning for computer vision applications. In2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC) 2021 (pp. 1183-1189).
75. Shi W, Rajkumar R. Point-gnn: Graph neural network for 3d object detection in a point cloud. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition 2020 (pp. 1711-1719).
76. Wu Y, Dai HN, Tang H. Graph neural networks for anomaly detection in industrial internet of things. IEEE Int Things J. 2021;9(12):9214–31.
77. Pitsik EN, et al. The topology of fMRI-based networks defines the performance of a graph neural network for the classification of patients with major depressive disorder. Chaos Solitons Fractals. 2023;167: 113041.
78. Liao W, Zeng B, Liu J, Wei P, Cheng X, Zhang W. Multi-level graph neural network for text sentiment analysis. Comput Electr Eng. 2021;92: 107096.
79. Kumar VS, Alemran A, Karras DA, Gupta SK, Dixit CK, Haralayya B. Natural Language Processing using Graph Neural Network for Text Classification. In2022 International Conference on Knowledge Engineering and Communication Systems (ICKES) 2022; (pp. 1-5).
80. Dara S, Srinivasulu CH, Babu CM, Ravuri A, Paruchuri T, Kilak AS, Vidyarthi A. Context-Aware auto-encoded graph neural model for dynamic question generation using NLP. ACM transactions on asian and low-resource language information processing. 2023.

81. Wu L, Cui P, Pei J, Zhao L, Guo X. Graph neural networks: foundation, frontiers and applications. InProceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2022; (pp. 4840-4841).
82. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G. The graph neural network model. IEEE Trans neural networks. 2008;20(1):61–80.
83. Cao P, Zhu Z, Wang Z, Zhu Y, Niu Q. Applications of graph convolutional networks in computer vision. Neural Comput Appl. 2022;34(16):13387–405.
84. You R, Yao S, Mamitsuka H, Zhu S. DeepGraphGO: graph neural network for large-scale, multispecies protein function prediction. Bioinformatics. 2021;37(Supplement_1):i262-71.
85. Long Y, et al. Pre-training graph neural networks for link prediction in biomedical networks. Bioinformatics. 2022;38(8):2254–62.
86. Wu Y, Gao M, Zeng M, Zhang J, Li M. BridgeDPI: a novel graph neural network for predicting drug–protein interactions. Bioinformatics. 2022;38(9):2571–8.
87. Kang C, Zhang H, Liu Z, Huang S, Yin Y. LR-GNN: a graph neural network based on link representation for predicting molecular associations. Briefings Bioinf. 2022;23(1):bbab513.
88. Wei X, Huang H, Ma L, Yang Z, Xu L. Recurrent Graph Neural Networks for Text Classification. in 2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS), 2020, pp. 91–97.
89. Schlichtkrull MS, De Cao N, Titov I. Interpreting graph neural networks for nlp with differentiable edge masking. arXiv Prepr. arXiv2010.00577, 2020.
90. Tu M, Huang J, He X, Zhou B. Graph sequential network for reasoning over sequences. arXiv Prepr. arXiv2004.02001, 2020.

Publisher's Note