Shahid Chamran University of Ahvaz

Journal of Advanced Mathematical Modeling
ISSN (print): 2251-8088, ISSN (on-line): 2645-6141
Vol. 16 No. 2 (2026), pp. 143-168.
DOI: 10.22055/jamm.2026.51276.2444

GRAPH NEURAL NETWORKS: MATHEMATICAL FOUNDATIONS, MODERN ARCHITECTURES, AND FUTURE DIRECTIONS

YAZDAN GOLZADEH, HASSAN MALEKI*

Communicated by: Saeid Alikhani

Abstract. Graph Neural Networks (GNNs) have become a central framework for learning from graph-structured data, with applications in social networks, recommendation systems, molecular science, knowledge graphs, and scientific discovery. This survey reviews the mathematical foundations, architectural evolution, and emerging directions of GNNs in a compact and unified form. We discuss message passing, spectral graph theory, graph convolution, attention-based aggregation, expressivity, stability, and the main limitations of deep graph models. Representative architectures, including recurrent GNNs, GCN, GraphSAGE, MPNN, GAT, heterogeneous GNNs, graph transformers, and graph language models, are compared from the viewpoint of their design principles, strengths, limitations, and application domains. The survey also summarizes benchmark datasets, practical modeling considerations, and future challenges such as scalability, over-smoothing, over-squashing, uncertainty estimation, foundation models, and graph–language integration. The aim is to provide a concise reference that connects theory, architecture, and practice in modern graph representation learning.

MSC(2020): 68T05, 68T07, 68T10, 68R10, 05C85.

Keywords: Graph Neural Networks, Non-Euclidean Data, Geometric Deep Learning, Message Passing, Graph Representation Learning.

Received: 29 November 2025, Accepted: 8 August 2026.

*Corresponding author.

1. Introduction

Deep learning has achieved remarkable success on Euclidean data such as images, audio, and text. Convolutional Neural Networks exploit the regular grid structure of images, while Recurrent Neural Networks and their variants model ordered sequences such as speech and natural language [24, 19]. These architectures, however, rely on assumptions that do not hold for graph-structured data. Graphs have irregular topology, variable size, unordered neighborhoods, and no global coordinate system [25]. Graph Neural Networks (GNNs) address these limitations by learning directly from graph connectivity and extending deep representation learning to non-Euclidean domains.

Unlike conventional neural networks, which mainly process independent observations, GNNs explicitly model relations among entities. In many scientific and engineering problems, the interaction pattern among entities is as informative as the attributes of the entities themselves. Molecular structures, protein interaction networks, citation networks, transportation systems, power grids, social networks, and knowledge graphs are typical examples where graph modeling is natural [8, 12, 17, 15, 28, 29, 34, 2, 43]. As a result, graph representation learning has become one of the core components of modern geometric deep learning.

GNNs support a broad range of tasks, including node classification, graph classification, link prediction, edge prediction, community detection, and graph generation [35]. Their effectiveness comes from iterative neighborhood aggregation: each node receives information from its neighbors, updates its representation, and gradually encodes both its own attributes and the surrounding graph topology [14]. This simple principle has led to a broad family of architectures, from early recurrent graph models to graph convolutional networks, attention-based GNNs, graph transformers, and recent graph language models.

1.1. **Challenges of Graph Learning.**

Graph learning differs from Euclidean learning in three fundamental aspects. First, graph topology is defined by combinatorial connectivity rather than spatial coordinates. Classical convolution cannot be directly applied because there is no fixed local grid or globally consistent neighborhood structure [6]. Second, graphs have variable size and irregular structure: different graphs may contain different numbers of vertices and edges, and different nodes may have neighborhoods of different

sizes. Third, the ordering of vertices is arbitrary. Relabeling node indices must not change the meaning of the graph or the output of the learning model.

These properties require learning algorithms that are local, topology-aware, and permutation invariant. GNNs satisfy these requirements through message passing and symmetric aggregation functions such as summation, mean, maximum, or attention-weighted aggregation. This makes them suitable for learning on graphs without imposing artificial Euclidean structure.

1.2. **Traditional Approaches to Graph Data Analysis.** Before GNNs, graph learning mainly relied on handcrafted structural features and graph kernels [6]. These approaches converted graphs into fixed-dimensional representations and then applied conventional machine learning algorithms. Although useful for small and well-defined problems, they often suffered from information loss, limited scalability, high computational cost, and dependence on expert-designed descriptors.

Early neural approaches extended recursive neural networks to graph-structured data by repeatedly propagating information between neighboring nodes [35]. These methods showed that graph representations could be learned from topology, but their fixed-point optimization procedures were expensive and difficult to scale. Modern GNNs replaced these recursive schemes with layer-wise neighborhood aggregation, making end-to-end graph representation learning practical for larger datasets.

1.3. **Scope and Contributions.** This survey provides a compact view of Graph Neural Networks by connecting their mathematical foundations, architectural evolution, benchmark evaluation, and emerging research directions. The main contributions are as follows.

- We review the mathematical principles of GNNs, including message passing, spectral graph theory, attention-based aggregation, expressivity, over-smoothing, and stability.
- We summarize representative architectures from recurrent GNNs and GCNs to graph transformers and graph language models.
- We compare major design trade-offs involving expressive power, scalability, robustness, interpretability, uncertainty estimation, and computational efficiency.
- We discuss representative benchmark datasets and application domains.

- We identify open challenges and future directions, including scalable graph learning, dynamic graphs, graph foundation models, and graph–language integration.

Figure 1 summarizes the scope of this survey by organizing Graph Neural Networks according to their foundational paradigms, representative architectural families, learning tasks, theoretical themes, and emerging research directions.

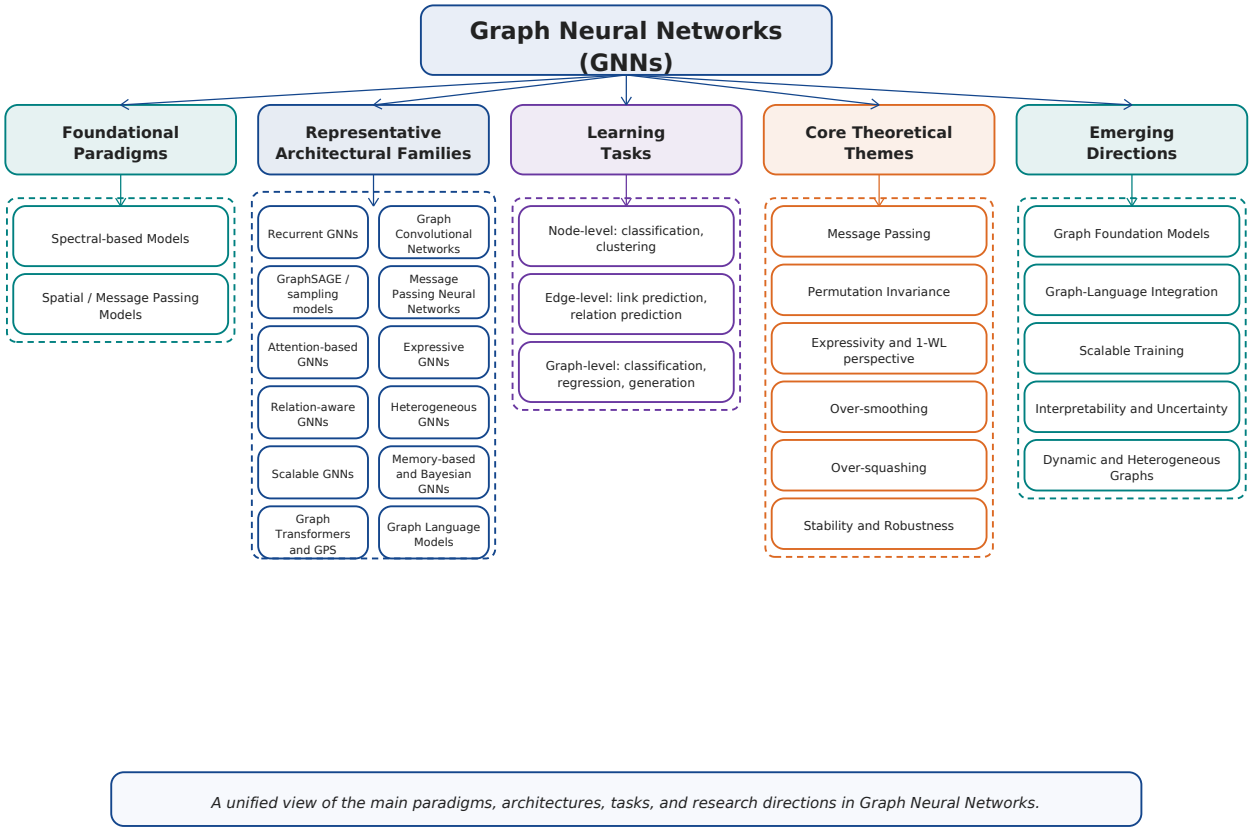


Figure 1. Taxonomy and landscape of Graph Neural Networks.

FIGURE 1. Taxonomy and landscape of Graph Neural Networks. The diagram organizes the field according to foundational paradigms, representative architectures, learning tasks, theoretical themes, and emerging research directions.

Table 1 shows that existing surveys usually emphasize one part of the graph learning landscape. Early surveys provide broad taxonomies of classical GNN architectures, while

more recent surveys focus on specialized topics such as graph transformers, graph foundation models, or the integration of large language models with graph learning. These studies are important, but they often treat mathematical foundations, architectural design, and practical modeling trade-offs as separate issues.

The present survey is positioned between these lines of work. Its purpose is not to provide an exhaustive catalogue of all GNN variants, nor to focus exclusively on a single emerging paradigm. Instead, it presents a compact and mathematically grounded view of GNNs as computational models for relational data. In this view, message passing, spectral graph theory, expressivity, architectural design, scalability, and graph-language integration are discussed as connected components of the same modeling framework.

1.4. **Literature Selection Strategy.** This article follows a structured narrative review rather than a formal systematic review. Representative studies were selected from IEEE Xplore, ACM Digital Library, SpringerLink, ScienceDirect, Web of Science, Scopus, and Google Scholar. Priority was given to influential journal articles and leading conference papers that introduced theoretical advances, representative GNN architectures, scalable learning methods, graph transformers, graph foundation models, and emerging research topics [38, 31].

The adopted selection procedure is summarized in Table 2. This table clarifies the type of literature considered, the databases searched, the publication period, the representative venues, and the main criteria used to identify relevant studies.

2. Fundamental Concepts

This section introduces only the concepts required for the rest of the survey. The purpose is not to provide a complete review of graph theory or deep learning, but to establish the notation and intuition used in GNN models.

2.1. **Graph Representation.** A graph is defined as

$$G = (V, E),$$

where V is the set of vertices and E is the set of edges connecting them [6]. Graphs may be directed or undirected, homogeneous or heterogeneous, and static or dynamic. These properties often coexist in real-world data. For example, a knowledge graph is typically

directed and multi-relational, while a social network may be large, sparse, dynamic, and heterogeneous.

Unlike Euclidean data, graphs have no fixed coordinate system, no canonical vertex ordering, and no uniform neighborhood structure. These properties make direct application of standard neural architectures difficult and motivate models that operate on connectivity patterns rather than spatial position.

2.2. **From Neural Networks to Graph Neural Networks.** Conventional neural networks were developed for regular domains. Feedforward Neural Networks process fixed-dimensional vectors, Convolutional Neural Networks exploit local spatial regularity, and Recurrent Neural Networks model sequential dependencies [19, 24, 25]. Graph data violate these assumptions because the neighborhood of each node is irregular and unordered.

GNNs extend deep learning to relational domains by incorporating graph topology into the learning process. Instead of learning from isolated feature vectors, they jointly model node attributes and the relationships among neighboring vertices. The representation of a node therefore depends on both its own features and the information propagated through its local graph context.

2.3. **Graph Neural Networks.** The modern formulation of GNNs was introduced by Scarselli *et al.* [35]. Since then, the field has evolved into a broad family of architectures, including graph convolutional networks, inductive neighborhood aggregation models, attention-based GNNs, graph transformers, and graph foundation models.

Despite their differences, most GNNs share the same computational principle. At each layer, nodes collect messages from their neighbors, aggregate them using a permutation-invariant function, and update their hidden states. Repeating this process allows information to propagate across the graph and enables node embeddings to encode progressively richer structural context. This principle, known as message passing, is the foundation of modern GNNs.

3. Mathematical Foundations of Graph Neural Networks

The mathematical study of Graph Neural Networks provides the language needed to understand why different architectures succeed or fail on graph-structured data. Although modern GNN models differ in their implementation details, most of them are built around

a small number of common principles: local information propagation, permutation-invariant aggregation, spectral or spatial graph filtering, expressive power, stability, and the behavior of repeated propagation over graph topology. This section reviews these principles in a unified form and shows how they connect the design of GNN architectures to the structural properties of graphs.

3.1. **Message Passing Framework.** Most modern GNN architectures can be expressed through the message-passing framework. Let $G = (V, E)$ be a graph with node features $X = \{x_v \mid v \in V\}$ and optional edge attributes e_{uv}. At layer t, node v receives messages from its neighborhood $\mathcal{N}(v)$ and updates its hidden representation by

$$m_v^{(t)} = \mathrm{AGGREGATE}^{(t)}\left(\left\{M^{(t)}\left(h_v^{(t-1)}, h_u^{(t-1)}, e_{uv}\right) : u \in \mathcal{N}(v)\right\}\right), \tag{3.1}$$

$$h_v^{(t)} = \mathrm{UPDATE}^{(t)}\left(h_v^{(t-1)}, m_v^{(t)}\right). \tag{3.2}$$

Here, $M^{(t)}$ constructs messages, $\mathrm{AGGREGATE}^{(t)}$ combines the received neighborhood information, and $\mathrm{UPDATE}^{(t)}$ produces the next node state. This formulation is broad enough to describe GCN, GraphSAGE, MPNN, GAT, GIN, and many recent variants; the main architectural differences lie in the choice of the message function, the aggregation rule, and the update mechanism [14].

A necessary structural requirement is permutation invariance. Since the ordering of neighbors is arbitrary, the aggregation function must satisfy

$$\mathrm{AGGREGATE}\left(\{h_1, h_2, \ldots, h_k\}\right) = \mathrm{AGGREGATE}\left(\{h_{\pi(1)}, h_{\pi(2)}, \ldots, h_{\pi(k)}\}\right) \tag{3.3}$$

for every permutation π of $\{1, \ldots, k\}$. Summation, mean, maximum, and attention-weighted aggregation are typical choices. This property ensures that a GNN depends on graph structure and node attributes, not on an arbitrary indexing of vertices.

The depth of a message-passing model determines the size of the receptive field. After one layer, $h_v^{(1)}$ depends on immediate neighbors; after L layers, $h_v^{(L)}$ may depend on vertices within L hops. This explains why message passing is effective for local relational learning, but it also clarifies its limitations: deeper propagation is needed for long-range information, while too much propagation may lead to over-smoothing or over-squashing.

3.2. **Spectral Graph Theory and Graph Convolution.** Spectral graph theory gives an early mathematical route from classical convolution to graph convolution [3, 6, 7]. For

an undirected graph with adjacency matrix A and degree matrix D, the normalized graph Laplacian is

$$L = I - D^{-1/2}AD^{-1/2}. \tag{3.4}$$

Because L is symmetric and positive semi-definite, it admits the eigendecomposition

$$L = U\Lambda U^T, \tag{3.5}$$

where the columns of U define graph Fourier modes and the diagonal entries of Λ represent graph frequencies. For a graph signal $x \in \mathbb{R}^{|V|}$, the graph Fourier transform is

$$\hat{x} = U^T x. \tag{3.6}$$

A spectral graph convolution can then be written as

$$y = U g_\theta(\Lambda) U^T x, \tag{3.7}$$

where $g_\theta(\Lambda)$ is a learnable spectral filter [6, 7].

Equation (3.7) is mathematically clear, but it has two important limitations. First, computing the eigendecomposition of L is expensive for large graphs. Second, the filter is tied to the eigenbasis of a specific graph, which weakens inductive transfer across different graph instances. ChebNet addresses these issues by approximating the spectral filter through Chebyshev polynomials [7]:

$$y = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})x, \qquad \tilde{L} = \frac{2}{\lambda_{\max}} L - I, \tag{3.8}$$

where $T_k(\cdot)$ is the k-th Chebyshev polynomial. This approximation avoids explicit eigendecomposition and yields a localized K-hop filtering operator. Kipf and Welling further simplified this idea into a first-order propagation rule, giving the standard GCN layer [23]. Thus, the historical movement from spectral convolution to spatial message passing can be understood as a move from exact graph Fourier filtering toward scalable localized graph operators.

3.3. **Attention-Based Message Passing.** In basic message passing, the influence of neighboring nodes is often fixed by degree normalization or a simple symmetric function. Attention-based GNNs replace this fixed aggregation with data-dependent weights. In

Graph Attention Networks, a compatibility score between a target node v and a neighbor u is computed by

$$e_{vu} = \mathrm{LeakyReLU}\left(a^T \left[Wh_v \,\|\, Wh_u\right]\right), \tag{3.9}$$

where W is a learnable linear map, a is a learnable attention vector, and $\|$ denotes concatenation. The scores are normalized over the neighborhood by

$$\alpha_{vu} = \frac{\exp(e_{vu})}{\sum_{w\in\mathcal{N}(v)} \exp(e_{vw})}, \qquad \sum_{u\in\mathcal{N}(v)} \alpha_{vu} = 1. \tag{3.10}$$

The resulting update is

$$h'_v = \sigma\left(\sum_{u\in\mathcal{N}(v)} \alpha_{vu} W h_u\right). \tag{3.11}$$

This operator remains local, but the contribution of each neighbor is learned from features rather than fixed by topology alone [41, 47].

Multi-head attention improves the robustness of the learned aggregation. With K attention heads, the update can be written as

$$h'_v = \mathrm{CONCAT}_{k=1}^{K}\left[\sigma\left(\sum_{u\in\mathcal{N}(v)} \alpha_{vu}^{(k)} W^{(k)} h_u\right)\right]. \tag{3.12}$$

Different heads may focus on different relational patterns in the same neighborhood. This makes attention useful for heterogeneous or noisy graphs, although the model still inherits the locality of message passing. Graph Transformers extend this idea by combining attention with positional or structural encodings so that long-range interactions can be modeled more directly [9, 32, 11].

3.4. **Expressive Power and the Weisfeiler–Lehman Perspective.** Expressivity asks what graph structures a GNN can distinguish. For standard message-passing GNNs, the key reference point is the one-dimensional Weisfeiler–Lehman test. A useful interpretation is that if two graphs cannot be distinguished by the 1-WL refinement procedure, then a broad class of message-passing GNNs with permutation-invariant neighborhood aggregation will also fail to distinguish them under the same initial features [46].

This limitation is not only theoretical; it explains why simple averaging or non-injective aggregation may lose structural information. The Graph Isomorphism Network was designed to reach the discriminative power of the 1-WL test by using an injective multiset

aggregation. Its layer can be written as

$$h_v^{(t)} = \mathrm{MLP}^{(t)}\left((1+\epsilon^{(t)})h_v^{(t-1)} + \sum_{u\in\mathcal{N}(v)} h_u^{(t-1)}\right). \tag{3.13}$$

The summation term preserves multiset information more faithfully than simple mean aggregation, while the multilayer perceptron increases the expressive capacity of the update. This perspective also clarifies why recent work studies higher-order GNNs, subgraph-based methods, and positional or structural encodings: these mechanisms attempt to move beyond the locality and symmetry constraints of standard message passing.

3.5. **Over-smoothing, Over-squashing, and Stability.** A second set of mathematical questions concerns the behavior of GNNs as depth increases. Over-smoothing occurs when repeated propagation makes node representations increasingly similar. To see this, consider the simplified linear propagation

$$H^{(L)} = \hat{A}^L X, \qquad \hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}, \tag{3.14}$$

where $\tilde{A} = A + I$. Under standard connectivity and aperiodicity assumptions, repeated diffusion tends toward a stationary subspace. In schematic form,

$$\lim_{L\to\infty} H^{(L)} = \mathbf{1}\pi^T X, \tag{3.15}$$

where π is the stationary distribution of the associated graph diffusion process. This limiting behavior explains why adding many propagation layers can reduce the discriminative power of node embeddings [26, 30].

Over-squashing is different. It occurs when information from a rapidly growing L-hop neighborhood is compressed into a fixed-dimensional vector. Even if representations do not become identical, long-range signals may be distorted or lost as they pass through narrow graph bottlenecks. This issue is especially important for tasks that depend on distant interactions. Recent analyses connect over-squashing to graph bottlenecks and curvature-based rewiring, suggesting that the topology of the input graph can directly limit the efficiency of message passing [1, 40].

Stability concerns the sensitivity of a GNN to perturbations in the graph. Real graph data may contain missing edges, spurious links, noisy attributes, or time-varying structure. If the propagation, transformation, and activation operators are Lipschitz continuous,

then the network can be analyzed as a composition of stable operators. A representative bound can be written schematically as

$$\|\mathrm{GNN}(A, X) - \mathrm{GNN}(A', X)\| \leq C_L \, \|A - A'\| \, \|X\| \,, \tag{3.16}$$

where A and A' are two nearby adjacency matrices and C_L depends on the depth and the Lipschitz constants of the layers. More refined stability results for graph neural networks are developed in [13]. For modeling purposes, the important point is that stability is not an optional property: it is essential when GNNs are used on noisy, incomplete, or dynamically changing graphs.

3.6. **GNNs as Nonlinear Graph Operators.** The preceding formulations can be combined into a compact operator view. Let

$$\mathcal{F}_\Theta : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d} \tag{3.17}$$

denote a parameterized graph propagation operator that depends on the graph structure and learnable parameters Θ. A single GNN layer can be written as

$$H^{(l+1)} = \mathcal{F}_\Theta \left(H^{(l)}, A \right) , \tag{3.18}$$

and an L-layer GNN corresponds to the composition

$$H^{(L)} = \mathcal{F}_\Theta^L \left(H^{(0)}, A \right) . \tag{3.19}$$

This view is useful because it places spectral convolution, spatial message passing, attention, and transformer-style graph models within the same mathematical language: all of them learn operators that propagate and transform signals over a graph.

It also explains why the theoretical issues above are connected. Expressivity depends on the class of graph operators that can be represented. Over-smoothing reflects the limiting behavior of repeated propagation. Over-squashing reflects the inability of a local operator to transmit too much distant information through narrow graph regions. Stability measures the sensitivity of the operator to perturbations. In this sense, the mathematical study of GNNs is not separate from architecture design; it provides the principles that explain why different models succeed, fail, or scale in different graph learning settings.

4. Types of GNN Models and Learning Tasks

GNNs can be classified from three complementary perspectives. The first is the graph structure: models may be designed for undirected, directed, heterogeneous, dynamic, or multi-relational graphs [39, 42]. The second is the learning mechanism: spectral convolution, spatial message passing, attention-based aggregation, transformer-based global attention, and probabilistic graph learning. The third is the prediction task: node-level, edge-level, and graph-level learning.

Node-level tasks include node classification and clustering, where the goal is to predict labels or communities for individual vertices. Edge-level tasks include link prediction and relation prediction. Graph-level tasks include molecular property prediction, graph classification, and graph generation. The choice of architecture depends strongly on the task, the available features, graph scale, and whether the model must generalize to unseen graphs.

5. Representative Graph Neural Network Architectures

The evolution of GNN architectures can be understood as a sequence of responses to specific limitations: early recurrent models introduced iterative information propagation, graph convolution made propagation efficient, sampling enabled inductive learning, attention improved adaptive aggregation, heterogeneous models handled relation types, and graph transformers addressed long-range dependencies.

The historical development of GNNs is summarized in Figure 2, which shows the transition from recurrent graph propagation and spectral convolution toward message passing, expressivity analysis, scalable learning, graph transformers, and graph foundation models.

Recurrent Graph Neural Networks. RecGNNs are the earliest graph neural architectures and formulate graph learning as a recurrent state-transition process [35]. They established the idea that local interactions among neighboring nodes can encode global structural information. Their main drawback is the need for iterative convergence, which makes training expensive and limits scalability.

Graph Convolutional Networks. GCNs provide an efficient layer-wise approximation of spectral graph convolution [23]. Their propagation rule can be written as

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right), \tag{5.1}$$

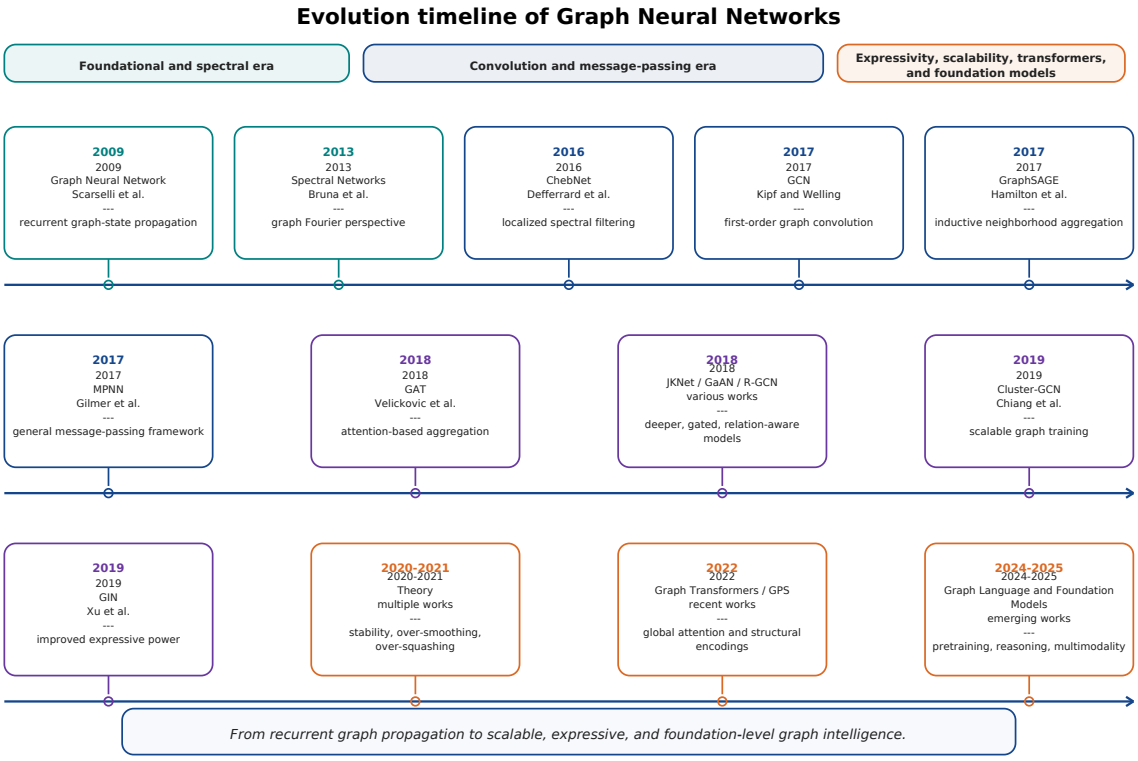


Figure 2. Evolution timeline of Graph Neural Networks.

FIGURE 2. Evolution timeline of Graph Neural Networks. The timeline highlights the conceptual transition from recurrent graph propagation and spectral convolution to spatial message passing, attention, expressivity analysis, scalable training, graph transformers, and graph foundation-model directions.

where $\tilde{A} = A + I$ and $\tilde{D}$ is its degree matrix. GCNs are simple and efficient, but their fixed local aggregation may limit expressivity and can lead to over-smoothing in deep models.

GraphSAGE. GraphSAGE introduces inductive neighborhood aggregation, allowing the model to generalize to unseen nodes or graphs [16]. Instead of using the full neighborhood, it samples neighbors and applies learnable aggregators such as mean, pooling, or LSTM-based aggregation. This makes GraphSAGE suitable for large and evolving graphs.

Message Passing Neural Networks. MPNNs provide a general framework rather than a single architecture [14]. They describe GNNs through message construction, neighborhood aggregation, and node-state updates. This formulation is particularly useful in molecular learning, where edge attributes and bond types can be incorporated into the message function.

Graph Attention Networks. GATs use attention coefficients to weight neighbors adaptively [41]. Unlike GCNs, which aggregate neighbors with fixed normalized weights, GATs learn the importance of each neighbor from data. They improve flexibility but may become computationally expensive on large dense graphs.

Relation-aware and Heterogeneous GNNs. Many real graphs contain multiple node types, edge types, or relation semantics. R-GCN extends message passing to multi-relational graphs by using relation-specific transformations. Heterogeneous models such as HAN and GTN further incorporate semantic relations and meta-path information, making them suitable for knowledge graphs and heterogeneous information networks [36, 42].

Memory-augmented and Bayesian GNNs. Memory-augmented GNNs introduce external memory modules to preserve long-term relational information, while Bayesian GNNs provide uncertainty estimation through probabilistic modeling [18, 22]. These directions are useful in settings where robustness and confidence estimation are important, but they increase model complexity and computational cost.

Graph Transformers and GPS. Local message passing struggles with long-range dependencies. Graph Transformers address this issue by using self-attention to allow interactions between distant nodes. However, global attention is computationally expensive on large graphs. Hybrid architectures such as GPS combine local message passing with global attention, seeking a balance between graph inductive bias and long-range modeling.

Graph Language Models. Recent graph-language models integrate graph representation learning with large language models [21, 33]. They may represent graphs as token sequences, align graph and text embeddings, or use language models for reasoning over graph-structured knowledge. This direction extends GNN research from task-specific representation learning toward graph generation, reasoning, and multimodal foundation models.

5.1. **Comparative Discussion.** No single GNN architecture is optimal for all settings. GCN remains attractive when simplicity and efficiency are priorities. GraphSAGE is

preferable for inductive and large-scale scenarios. GAT improves adaptive aggregation but may be costly on dense graphs. Relation-aware and heterogeneous models are suitable when edge types or semantic relations matter. Graph Transformers improve long-range modeling but require careful control of computational complexity. The appropriate choice depends on graph scale, feature availability, topology, task type, and computational constraints.

The qualitative trade-offs among representative architectures are summarized in Figure 3. The figure complements Table 3 by emphasizing that architectural choice depends on graph scale, relation type, long-range dependency requirements, robustness, and computational budget.

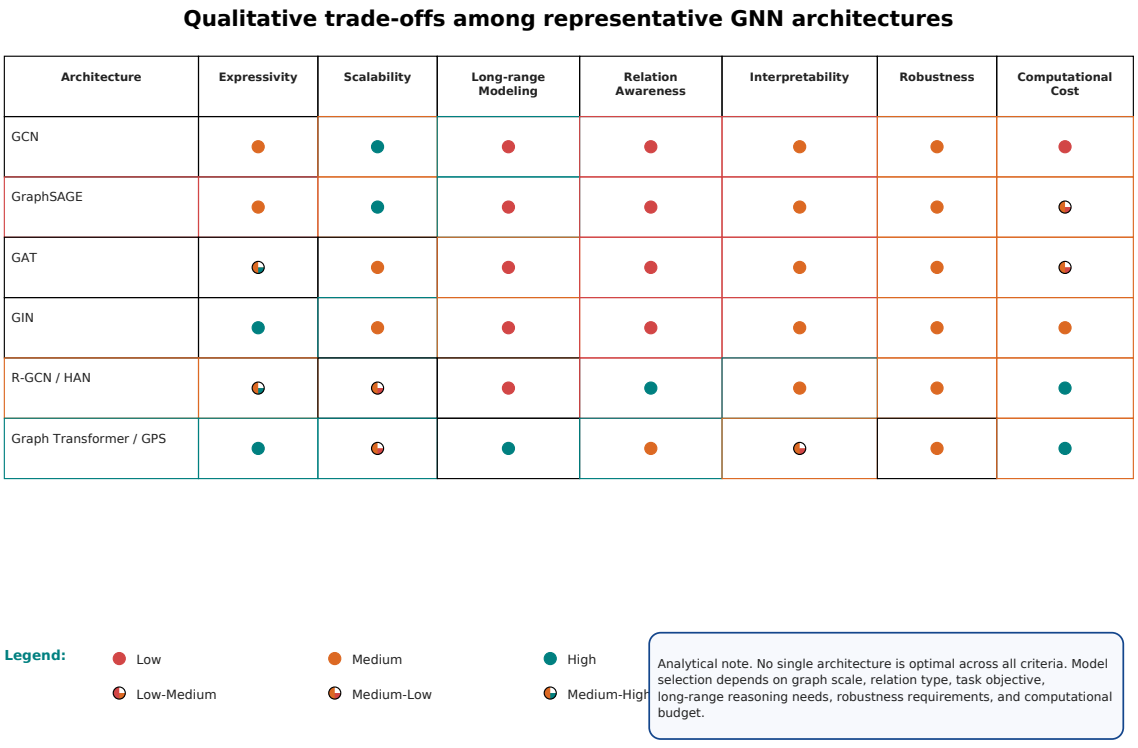


Figure 4. Qualitative trade-offs among representative GNN architectures.

FIGURE 3. Qualitative trade-offs among representative GNN architectures. The symbols indicate approximate relative levels rather than benchmark scores. For computational cost, a higher level indicates greater computational burden.

Table 4 reports representative benchmark results collected from original model papers and widely used benchmark studies. The reported values should be interpreted as indicative numerical references rather than as a unified leaderboard, since the datasets, splits, metrics, parameter budgets, and training protocols are not identical across all sources. To avoid unsupported comparison, the table explicitly specifies the task, evaluation metric, reported values, and the corresponding source or protocol for each dataset. In this sense, Table 4 provides a compact numerical orientation for commonly used graph-learning benchmarks and representative GNN architectures.

6. Emerging Applications and Practical Limitations

GNNs are now used in a wide range of domains. In molecular science, atoms and bonds are modeled as nodes and edges, enabling molecular property prediction, drug discovery, and reaction modeling. In recommendation systems, users and items form bipartite or heterogeneous graphs. In knowledge graphs, relation-aware GNNs support entity classification, link prediction, and reasoning. In social and citation networks, GNNs are widely used for node classification, community detection, and influence analysis.

Despite these successes, practical deployment remains challenging. Large graphs require sampling or mini-batch training. Heterophilous graphs may violate the assumption that neighboring nodes have similar labels. Dynamic graphs require temporal modeling rather than static aggregation. In safety-sensitive domains, uncertainty estimation and robustness to graph perturbations are essential. These limitations continue to motivate new architectures and training strategies.

7. Challenges and Future Directions

7.1. **Over-smoothing and Over-squashing.**

Over-smoothing makes node embeddings increasingly similar as the number of layers grows. Over-squashing occurs when information from many distant nodes is compressed into fixed-size representations. Both problems limit deep message passing and motivate residual connections, normalization, Jumping Knowledge-style layer aggregation [45], rewiring, curvature-based methods, and hybrid attention architectures.

7.2. **Scalability.** Many real-world graphs contain millions or billions of edges. Full-batch training is often infeasible, and global attention is expensive. Scalable GNN research therefore focuses on sampling, clustering, subgraph training, sparse attention, and distributed graph learning [17, 5].

7.3. **Generalization and Expressivity.** The expressive power of GNNs remains limited by local aggregation and graph isomorphism constraints. Future work must improve structural discrimination while preserving efficiency and robustness. Generalization across datasets, domains, and graph sizes is especially important for graph foundation models.

7.4. **Interpretability and Uncertainty.** GNNs are increasingly used in scientific and high-stakes settings, where predictions must be reliable and interpretable. Better explanation methods, uncertainty-aware learning, calibration, and robustness guarantees are needed for trustworthy graph learning.

7.5. **Graph Foundation Models and Graph–Language Integration.** Large-scale pretraining is changing the scope of graph learning. Graph foundation models aim to learn transferable representations across tasks and domains, while graph–language models connect structured graph reasoning with natural language understanding. These directions are promising, but they require careful treatment of scalability, data quality, evaluation protocols, and domain transfer.

8. Conclusion

This survey reviewed Graph Neural Networks from a unified mathematical and modeling perspective. Rather than treating GNN architectures as isolated model families, the paper organized them around a common computational principle: the propagation and transformation of graph signals through message passing, spectral filtering, attention-based aggregation, and nonlinear graph operators. This viewpoint clarifies how classical graph convolution, spatial aggregation, attention mechanisms, graph transformers, and recent graph–language models can be understood as different realizations of graph-based representation learning.

A central conclusion of this review is that the development of GNNs is governed by a set of persistent trade-offs rather than by a single dominant architecture. Local message-passing models such as GCN, GraphSAGE, GAT, and GIN provide efficient and

topology-aware learning mechanisms, but their locality creates limitations in long-range reasoning, over-smoothing, and over-squashing. Transformer-based and foundation-model approaches improve global interaction modeling and transferability, but introduce new challenges in scalability, structural encoding, data requirements, and computational cost. Similarly, increasing expressive power often requires more complex aggregation schemes, higher-order structures, or positional information, which may reduce simplicity and efficiency.

From a theoretical perspective, the most important insight is that several practical limitations of GNNs arise from the mathematical structure of repeated graph propagation. Over-smoothing reflects the tendency of iterative diffusion to reduce representation diversity, while over-squashing reveals the difficulty of transmitting distant information through narrow graph bottlenecks. Stability, expressivity, and robustness are therefore not peripheral issues; they are central design principles for reliable graph learning. This observation reinforces the need to connect architectural innovation with mathematical analysis.

For future research, progress in graph representation learning should focus on models that balance expressivity, scalability, robustness, interpretability, and generalization. Promising directions include scalable graph transformers, graph foundation models, graph–language integration, uncertainty-aware GNNs, dynamic and heterogeneous graph learning, and methods that go beyond the limitations of standard 1-WL message passing. At the same time, stronger evaluation protocols and clearer benchmark settings are needed to make architectural comparisons more reliable. Overall, GNN research is moving from local neighborhood aggregation toward broader, more transferable, and theoretically informed graph learning systems.

References

[1] Alon, U. and Yahav, E., 2020. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*. doi: 10.48550/arXiv.2006.05205.

[2] Battaglia, P.W., Pascanu, R., Lai, M., Rezende, D.J. and Kavukcuoglu, K., 2016. Interaction networks for learning about objects, relations and physics. *arXiv preprint arXiv:1612.00222*. doi: 10.48550/arXiv.1612.00222.

[3] Bruna, J., Zaremba, W., Szlam, A. and LeCun, Y., 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*. doi: 10.48550/arXiv.1312.6203.

[4] Chen, C., Wu, Y., Dai, Q., Zhou, H.-Y., Xu, M., Yang, S., Han, X. and Yu, Y., 2024. A survey on graph neural networks and graph transformers in computer vision: A task-oriented perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12), pp.10297–10318. doi: 10.1109/TPAMI.2024.3445463.

[5] Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S. and Hsieh, C.-J., 2019. Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp.257–266. doi: 10.1145/3292500.3330925.

[6] Chung, F.R.K., 1997. *Spectral Graph Theory*. Providence, RI: American Mathematical Society. doi: 10.1090/cbms/092.

[7] Defferrard, M., Bresson, X. and Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In: *Advances in Neural Information Processing Systems*, 29, pp.3844–3852. doi: 10.5555/3157382.3157527.

[8] Duvenaud, D.K., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A. and Adams, R.P., 2015. Convolutional networks on graphs for learning molecular fingerprints. In: *Advances in Neural Information Processing Systems*, 28, pp.2224–2232. doi: 10.5555/2969442.2969488.

[9] Dwivedi, V.P. and Bresson, X., 2020. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*. doi: 10.48550/arXiv.2012.09699.

[10] Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y. and Bresson, X., 2020. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*. doi: 10.48550/arXiv.2003.00982.

[11] Dwivedi, V.P., Luu, A.T., Laurent, T., Bengio, Y. and Bresson, X., 2021. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*. doi: 10.48550/arXiv.2110.07875.

[12] Fout, A., Byrd, J., Shariat, B. and Ben-Hur, A., 2017. Protein interface prediction using graph convolutional networks. In: *Advances in Neural Information Processing Systems*, 30, pp.6530–6539. doi: 10.5555/3295222.3295399.

[13] Gama, F., Ribeiro, A. and Bruna, J., 2020. Stability properties of graph neural networks. *IEEE Transactions on Signal Processing*, 68, pp.5680–5695. doi: 10.1109/TSP.2020.3026980.

[14] Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O. and Dahl, G.E., 2017. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*. doi: 10.48550/arXiv.1704.01212.

[15] Hamaguchi, T., Oiwa, H., Shimbo, M. and Matsumoto, Y., 2017. Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, pp.1802–1808. doi: 10.24963/ijcai.2017/250.

[16] Hamilton, W.L., 2020. *Graph Representation Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. San Rafael, CA: Morgan & Claypool Publishers. doi: 10.2200/S01045ED1V01Y202009AIM046.

[17] Hamilton, W.L., Ying, Z. and Leskovec, J., 2017. Inductive representation learning on large graphs. In: *Advances in Neural Information Processing Systems*, 30, pp.1025–1035. doi: 10.5555/3294771.3294869.

[18] Hasanzadeh, A., Hajiramezanali, E., Boluki, S., Zhou, M., Duffield, N., Narayanan, K. and Qian, X., 2020. Bayesian graph neural networks with adaptive connection sampling. *arXiv preprint arXiv:2006.04064*. doi: 10.48550/arXiv.2006.04064.

[19] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. *Neural Computation*, 9(8), pp.1735–1780. doi: 10.1162/neco.1997.9.8.1735.

[20] Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M. and Leskovec, J., 2020. Open graph benchmark: Datasets for machine learning on graphs. In: *Advances in Neural Information Processing Systems*, 33, pp.22118–22133. doi: 10.5555/3495724.3497579.

[21] Jin, B., Liu, G., Han, C., Jiang, M., Ji, H. and Han, J., 2024. Large language models on graphs: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 36(12), pp.8622–8642. doi: 10.1109/TKDE.2024.3469578.

[22] Khasahmadi, A.H., Hassani, K., Moradi, P., Lee, L. and Morris, Q., 2020. Memory-based graph networks. *arXiv preprint arXiv:2002.09518*. doi: 10.48550/arXiv.2002.09518.

[23] Kipf, T.N. and Welling, M., 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*. doi: 10.48550/arXiv.1609.02907.

[24] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, 25, pp.1097–1105. doi: 10.5555/2999134.2999257.

[25] LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. *Nature*, 521(7553), pp.436–444. doi: 10.1038/nature14539.

[26] Li, Q., Han, Z. and Wu, X.-M., 2018. Deeper insights into graph convolutional networks for semi-supervised learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), pp.3538–3545. doi: 10.1609/aaai.v32i1.11604.

[27] Liu, J., Yang, C., Lu, Z., Chen, J., Li, Y., Zhang, M., Bai, T., Fang, Y., Sun, L., Yu, P.S. and Shi, C., 2025. Graph foundation models: Concepts, opportunities and challenges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6), pp.5023–5044. doi: 10.1109/TPAMI.2025.3548729.

[28] Moallemy-Oureh, A., Beddar-Wiesing, S., Nagel, Y., Nather, R. and Thomas, J.M., 2022. Marked neural spatio-temporal point process involving a dynamic graph neural network. *arXiv preprint arXiv:2206.03469*. doi: 10.48550/arXiv.2206.03469.

[29] Monti, F., Boscaini, D., Masci, J., Rodolà, E., Svoboda, J. and Bronstein, M.M., 2017. Geometric deep learning on graphs and manifolds using mixture model CNNs. In: *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.5425–5434. doi: 10.1109/CVPR.2017.576.

[30] Oono, K. and Suzuki, T., 2019. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*. doi: 10.48550/arXiv.1905.10947.

[31] Paul, J. and Criado, A.R., 2020. The art of writing literature review: What do we know and what do we need to know? *International Business Review*, 29(4), Art. no.101717. doi: 10.1016/j.ibusrev.2020.101717.

[32] Rampášek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G. and Beaini, D., 2022. Recipe for a general, powerful, scalable graph transformer. In: *Advances in Neural Information Processing Systems*, 35, pp.14501–14515. doi: 10.52202/068431-1054.

[33] Ren, X., Tang, J., Yin, D., Chawla, N.V. and Huang, C., 2024. A survey of large language models for graphs. In: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, pp.6616–6626. doi: 10.1145/3637528.3671460.

[34] Sanchez-Gonzalez, A., Heess, N., Springenberg, J.T., Merel, J., Riedmiller, M., Hadsell, R. and Battaglia, P., 2018. Graph networks as learnable physics engines for inference and control. *arXiv preprint arXiv:1806.01242*. doi: 10.48550/arXiv.1806.01242.

[35] Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M. and Monfardini, G., 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1), pp.61–80. doi: 10.1109/TNN.2008.2005605.

[36] Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I. and Welling, M., 2018. Modeling relational data with graph convolutional networks. In: *Proceedings of the 15th Extended Semantic Web Conference (ESWC)*, pp.593–607. doi: 10.1007/978-3-319-93417-4_38.

[37] Shehzad, A., Xia, F., Abid, S., Peng, C., Yu, S., Zhang, D. and Verspoor, K., 2026. Graph transformers: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 37(7), pp.3052–3071. doi: 10.1109/TNNLS.2025.3646122.

[38] Snyder, H., 2019. Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, 104, pp.333–339. doi: 10.1016/j.jbusres.2019.07.039.

[39] Tong, Z., Liang, Y., Sun, C., Rosenblum, D.S. and Lim, A., 2020. Directed graph convolutional network. *arXiv preprint arXiv:2004.13970*. doi: 10.48550/arXiv.2004.13970.

[40] Topping, J., Di Giovanni, F., Chamberlain, B.P., Dong, X. and Bronstein, M.M., 2021. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*. doi: 10.48550/arXiv.2111.14522.

[41] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y., 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*. doi: 10.48550/arXiv.1710.10903.

[42] Wang, X., Ji, H., Shi, C., Wang, B., Cui, P., Yu, P.S. and Ye, Y., 2019. Heterogeneous graph attention network. In: *Proceedings of the World Wide Web Conference*, pp.2022–2032. doi: 10.1145/3308558.3313562.

[43] Watters, N., Tacchetti, A., Weber, T., Pascanu, R., Battaglia, P. and Zoran, D., 2017. Visual interaction networks. *arXiv preprint arXiv:1706.01433*. doi: 10.48550/arXiv.1706.01433.

[44] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. and Yu, P.S., 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1), pp.4–24. doi: 10.1109/TNNLS.2020.2978386.

[45] Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-I. and Jegelka, S., 2018. Representation learning on graphs with jumping knowledge networks. *arXiv preprint arXiv:1806.03536*. doi: 10.48550/arXiv.1806.03536.

[46] Xu, K., Hu, W., Leskovec, J. and Jegelka, S., 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*. doi: 10.48550/arXiv.1810.00826.

[47] Zhang, J., Shi, X., Xie, J., Ma, H., King, I. and Yeung, D.-Y., 2018. GaAN: Gated attention networks for learning on large and spatiotemporal graphs. *arXiv preprint arXiv:1803.07294*. doi: 10.48550/arXiv.1803.07294.

[48] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M., 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1, pp.57–81. doi: 10.1016/j.aiopen.2021.01.001.

TABLE 1. Comparison with representative survey articles on Graph Neural Networks and recent graph learning paradigms. The comparison highlights the relative emphasis of each survey and clarifies the positioning of the present work.

Survey	**Year**	**Main Focus**	**Math. Foundations**	**Classical GNNs**	**Graph Transformers**	**LLM/GFM Topics**	**Main Limitation**
Wu et al. [44]	2021	General taxonomy of GNN models	Moderate	Strong	Limited	Not covered	Limited coverage of recent transformer-based and foundation-model directions.
Zhou et al. [48]	2020	Methods and applications of GNNs	Basic	Strong	Not covered	Not covered	Mainly application-oriented and less focused on mathematical formulation.
Chen et al. [4]	2024	GNNs and graph transformers in computer vision	Moderate	Moderate	Strong	Not covered	Task-oriented scope, mainly focused on computer vision.
Shehzad et al. [37]	2024	Graph transformers	Moderate	Limited	Strong	Limited	Focused on transformer-based models rather than the full GNN landscape.
Liu et al. [27]	2025	Graph foundation models	Moderate	Moderate	Moderate	Strong	Focused mainly on foundation-model concepts and opportunities.
Jin et al. [21]	2024	Large language models on graphs	Basic	Limited	Moderate	Strong	Focused on LLM–graph integration rather than classical GNN theory.
Ren et al. [33]	2024	Large language models for graphs	Basic	Limited	Moderate	Strong	Focused mainly on LLM-based graph learning frameworks.
This survey	**2025**	**Unified mathematical and modeling perspective**	**Strong**	**Strong**	**Moderate**	**Moderate**	**Balances mathematical foundations, architectural evolution, modeling trade-offs, and emerging directions in a compact framework.**

TABLE 2. Summary of the literature selection strategy adopted in this survey.

Criterion	**Description**
Literature Type	Peer-reviewed journal articles and leading international conference papers.
Databases	IEEE Xplore, ACM Digital Library, SpringerLink, ScienceDirect, Web of Science, Scopus, and Google Scholar.
Publication Period	Foundational studies together with recent advances published up to 2025.
Representative Venues	NeurIPS, ICML, ICLR, KDD, AAAI, IJCAI, CVPR, ICCV, and leading journals in machine learning, artificial intelligence, graph learning, and data mining.
Selection Priorities	Theoretical foundations, representative architectures, scalability, graph transformers, foundation models, practical applications, and emerging directions.

TABLE 3. Comparative analysis of representative Graph Neural Network architectures.

Architecture	**Core Design Principle**	**Major Strengths**	**Main Limitations**	**Representative Domains**
RecGNN	Recursive state propagation	Original graph neural paradigm	High cost and convergence issues	Node classification
GCN	Localized graph convolution	Simple and efficient	Limited receptive field; over-smoothing	Citation networks
GraphSAGE	Sampled inductive aggregation	Generalizes to unseen nodes	Sampling sensitivity	Large-scale networks
MPNN	General message passing	Flexible and widely applicable	Locality limitations	Molecular learning
GAT	Attention-based aggregation	Adaptive neighbor weighting	Higher cost on large graphs	Social and citation networks
R-GCN	Relation-specific propagation	Multi-relational modeling	Parameter growth	Knowledge graphs
HAN/GTN	Heterogeneous semantic modeling	Captures meta-path semantics	Complexity and meta-path dependence	Heterogeneous networks
Graph Transformer/GPS	Global attention with structural encoding	Long-range dependency modeling	Scalability challenges	Large graphs, molecules
GLM	Graph–language modeling	Reasoning and generation	Data and computation demands	Knowledge and molecular graphs

TABLE 4. Representative reported benchmark results for selected Graph Neural Network models. The values are taken from the original model papers or standard benchmark studies. They are intended to provide an indicative comparison across commonly used datasets, not a unified leaderboard under identical experimental settings.

Dataset	**Task / Domain**	**Metric**	**Representative reported results**	**Protocol / Source note**
Cora	Node classification / citation network	Accuracy (%)	GCN: 81.5; GAT: 83.0 ± 0.7	Transductive semi-supervised setting on the standard citation-network split reported in [23, 41].
Citeseer	Node classification / citation network	Accuracy (%)	GCN: 70.3; GAT: 72.5 ± 0.7	Transductive semi-supervised setting on the standard citation-network split reported in [23, 41].
PubMed	Node classification / citation network	Accuracy (%)	GCN: 79.0; GAT: 79.0 ± 0.3	Transductive semi-supervised setting on the standard citation-network split reported in [23, 41].
PPI	Multi-label node classification / biological networks	Micro-F1	GraphSAGE-LSTM: 0.612; GAT: 0.973 ± 0.002	Inductive setting with unseen test graphs. The GraphSAGE result is reported in [17]; the GAT result is reported in [41].
Reddit	Node classification / social network	F1	GraphSAGE-LSTM: 0.954	Large-scale inductive node-classification benchmark reported in the GraphSAGE paper [17].
ogbn-proteins	Multi-label node classification / protein association network	ROC-AUC (%)	GCN: 72.51 ± 0.35; GraphSAGE: 77.68 ± 0.20	Official OGB evaluation setting with standardized split and metric reported in [20].
ZINC	Graph regression / molecular graphs	MAE ↓	GCN: 0.278 ± 0.003; GAT: 0.384 ± 0.007; GIN: 0.526 ± 0.051; GPS: 0.070 ± 0.002	ZINC-12K molecular regression benchmark. GCN, GAT, and GIN values follow the Benchmarking GNNs protocol [10]; GPS follows [32].

Yazdan Golzadeh
Tehran, Iran.
Email: yazdan.golzadeh1984@gmail.com

Hassan Maleki
Department of Mathematics,
Faculty of Mathematical Sciences and Statistics,
Department of Artificial Intelligence and Data Science,
Faculty of Interdisciplinary Science and Technology,
Malayer University,
Malayer, Iran.
Email: hmaleki@malayeru.ac.ir

©2026 Shahid Chamran University of Ahvaz, Ahvaz, Iran. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution-NonComertial 4.0 International (CC BY-NC 4.0 license) (http://creativecommons.org/licenses/by-nc/4.0/).