Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.086118

ARTICLE

Sparse Structural Knowledge Enhanced Graph Neural Networks for Anomaly Detection in Social Networks

Zehan Li[1], Yingyi Li[2,*], Zhiwei Tang[3], Xuemeng Zhai[3], Jiandong Liang[1] and Guangmin Hu[3]

[1]School of Resources and Environment, University of Electronic Science and Technology of China, Chengdu, China
[2]College of Computer Science and Cyber Security (Pilot Software College), Chengdu University of Technology, Chengdu, China
[3]School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China

*Corresponding Author: Yingyi Li. Email: liyingyi@cdut.edu.cn

Received: 25 May 2026; Accepted: 09 July 2026; Published: 13 August 2026

ABSTRACT: Social network platforms have become primary channels for information dissemination, yet they are increasingly exploited by anomalous users such as bots, fake accounts, and coordinated disinformation spreaders. These malicious actors manipulate public opinion, spread misinformation and undermine platform integrity, posing severe threats to the security of the online ecosystem. Accurate detection of such users is challenging because they often organize into sophisticated high-order connection patterns that extend beyond local neighborhoods. Existing methods address this by either injecting predefined motifs as handcrafted features, which lack flexibility to discover unknown patterns, or employing higher-order Graph neural networks (GNNs) at prohibitive costs. Crucially, neither method treats structural information as learnable knowledge that can be automatically acquired from data and explicitly represented. To bridge this gap, we propose SparseGNN, a structural-knowledge-enhanced framework for anomalous user detection. It regards atomic subgraph patterns as fundamental, learnable units of structural knowledge. This framework is concatenated with original node features and fed into any standard GNN, without modifying the backbone architecture. Experiments on real-world datasets demonstrate that SparseGNN improves the accuracy and F1-score of standard GNNs for anomalous users detection without requiring predefined patterns, while maintaining linear complexity. Because the learned atomic patterns capture global high-order topology, the resulting structural knowledge representation is inherently less sensitive to localized edge perturbations, incidentally conferring improved stability under adversarial structural attacks.

KEYWORDS: Graph neural networks; anomalous user detection; sparse representation

1 Introduction

Social network platforms have become primary channels for large-scale information exchange. However, they are increasingly infiltrated by various anomalous users [1,2], such as bots, fake accounts [3], coordinated disinformation spreaders [4], and spammers [5,6]. These actors not only degrade platform ecosystems, but also manipulate public opinion, amplify misinformation [7], and pose threats to cyberspace security [8] and social stability [9]. These anomalous users exhibit distinctive high-order connection patterns that deviate significantly from normal users [10–12], such as forming abnormally dense subgraphs, star-like hub structures, or tightly coupled communities. When such high-order structural knowledge remains under exploited, detection models inevitably develop structural blind spots, conflating anomalous subgraph signatures with routine local connectivity [13].

Graph neural networks (GNNs) have emerged as the dominant approach for social network analysis due to their ability to model graph-structured data [14–16]. Yet standard message-passing GNNs, such as GCN [17], GAT [18], and GraphSAGE [19], aggregate information only from immediate neighbors. This local aggregation mechanism inherently limits their capacity to perceive high-order structural patterns extending beyond a node's direct vicinity [20]. To overcome this limitation, researchers have explored two main directions. The first involves subgraph-enhanced methods, where predefined substructures (e.g., triangles, cycles, stars) are counted and injected as handcrafted features into GNNs [21]. Although these methods improve structural awareness to some extent, they rely on fixed, domain-specific patterns and lack the flexibility to discover novel or dataset-specific anomalous structures. The second direction comprises higher-order GNNs, which explicitly model tuples of nodes or multi-hop neighborhoods to capture high-order interactions [22,23]. While theoretically more expressive, their computational complexity often grows rapidly, making them impractical for large-scale social networks. More importantly, both directions fail to provide a mechanism that manifests structural knowledge as concrete vectors, decomposes it into semantically meaningful atomic primitives, and automatically discovers dataset-specific anomalous patterns lying beyond human preset imagination, all while incurring only linear computational overhead. This gap motivates SparseGNN, which treats atomic subgraph patterns as learnable, decomposable, and explicitly representable units of structural knowledge.

In this paper, we define structural knowledge as learnable, decomposable, and explicitly representable knowledge about high-order connection patterns in graphs. The fundamental semantic units of such knowledge are atomic subgraph patterns, frequent local topological primitives that characterize how normal or anomalous users connect to their surroundings. Unlike hand-crafted statistical counts or implicit neural parameters, these atomic patterns are automatically discovered from data and carry explicit topological semantics. Based on this view, we construct a structural knowledge descriptor for each node by representing its local neighborhood as a sparse recombination of atomic units, thereby establishing an explicit mapping from raw graph topology to structured knowledge vectors. Building upon this notion, we propose SparseGNN, a Structural-knowledge-enhanced Graph Neural Network framework for anomalous user detection. The framework learns a dictionary of atomic subgraph patterns from the data, and encodes each node's local structure as a sparse weighted combination of these patterns to produce a compact descriptor. This descriptor captures global high-order topological information beyond the reach of local neighborhood aggregation, and is concatenated with original node features to augment any standard GNN backbone for end-to-end detection. The design is plug-and-play, preserving the base GNN architecture while incurring only linear computational overhead, which makes it readily applicable to diverse social network anomaly detection scenarios.

The main contributions of this paper are summarized as follows:

- Automatic acquisition of structural knowledge. We propose an atomic structure extraction process based on label-balanced sampling. Training labels are used exclusively to construct a balanced node subset, ensuring that both normal and anomalous structures are equally represented in the input data. The framework then automatically discovers a dictionary of atomic subgraph patterns from global topology without predefined motifs or manual feature engineering, preventing dictionary bias toward the majority normal class.
- Explicit representation of structural knowledge. We design a structural knowledge descriptor based on atom importance weights, which explicitly encodes a node's local high-order topology into a compact vector. This descriptor captures global structural information while preserving atom-level interpretability, achieving an explicit mapping from graph structure to structured knowledge representation.

- Structural-knowledge-enhanced anomalous user detection. We propose SparseGNN, a plug-and-play GNN enhancement framework. By fusing structural knowledge descriptors with original node features, it improves standard GNNs' perception of high-order anomalous structures while maintaining linear time complexity and the standard GNN training pipeline.
- Experimental validation and characteristic analysis. We conduct experiments on real-world social network datasets, validating the effectiveness of the acquired structural knowledge for anomaly detection. Further analysis reveals that because atomic patterns encode global topology, the knowledge enhancement strategy incidentally reduces sensitivity to localized adversarial perturbations, providing empirical evidence for the robustness of structural knowledge representation.

The remainder of this paper is organized as follows. Section 2 reviews related work on subgraph-aware GNNs, graph anomaly detection, and GNN robustness. Section 3 details the proposed SparseGNN framework, including structural knowledge acquisition, representation, and the enhancement mechanism. Section 4 presents experimental results on bot detection and adversarial robustness analysis. Section 5 concludes the paper.

2 Related Work

The local aggregation mechanism of standard message-passing GNNs inherently limits their ability to perceive high-order structural patterns. To enhance the structural awareness of GNNs, researchers have explored various directions.

2.1 Subgraph Aware Graph Neural Networks

The core idea of subgraph structure augmentation methods is to compensate for the limitations of local GNN aggregation by injecting subgraph-level structural information [24]. This direction was initially inspired by graph kernel methods. Classical graph kernels, such as the Weisfeiler-Lehman kernel and graphlet kernels, measure similarity between graphs by enumerating or counting predefined substructures (e.g., paths, cycles, cliques) [25]. Inspired by these approaches, researchers have integrated such ideas into GNN frameworks, forming an important branch of subgraph structure augmentation.

A representative work is the Graph Substructure Network (GSN) [26], which pre-specifies a set of subgraph patterns (e.g., triangles, cycles, stars), counts their isomorphism occurrences in each node's neighborhood, and injects these counts as structural features into the message-passing process. While GSN allows automated pattern generation, it still relies on a fixed, pre-computed pattern set and cannot learn task-specific patterns.

Another line of research enhances structural awareness through automatic encoding or sampling. The Nested Graph Neural Network (NGNN) [27] proposes a nested framework that extracts rooted subgraphs around individual nodes and processes them through hierarchical graph neural architectures, thereby enriching node representations with local structural information. NGNN is capable of encoding richer local structural information while introducing only a constant-factor higher time complexity. Nevertheless, NGNN requires processing a rooted subgraph for each node independently, which can incur significant overhead, although approximations are introduced to mitigate this issue.

Another prominent direction to enhance expressivity is the family of higher-order graph neural networks. These methods, including k-GNNs [22] that operate on tuples of nodes and leverage k-dimensional Weisfeiler-Lehman (WL) tests, invariant graph networks [23] built upon permutation-equivariant tensor architectures, and higher-order architectures linking graph isomorphism testing to GNN expressivity [28], capture high-order interactions beyond pairwise edges. While theoretically capable of distinguishing

non-isomorphic graphs that 1-WL cannot, they suffer from high computational complexity. Sparse implementations (e.g., considering only non-zero entries or using locality-sensitive hashing) can reduce the constant and lower the effective exponent, but the complexity remains significantly higher than that of standard message-passing GNNs. Moreover, these methods are primarily designed for small benchmark datasets (e.g., MUTAG, PROTEINS) and their message-passing schemes deviate from standard GNN architectures, making them difficult to integrate as a plug-in enhancement for existing models.

In summary, existing structure-aware GNNs fall into three categories, each with intrinsic limitations for anomalous user detection in social networks. Predefined-pattern methods rely on handcrafted motifs or fixed enumeration, lacking flexibility, decomposability, and the ability to discover dataset-specific structures. Automatic encoding or sampling methods produce implicit subgraph encodings that cannot be decomposed at the atomic level. Higher-order GNNs offer strong theoretical expressivity but embed topology into implicit network parameters, precluding explicit per-node structural descriptors, and suffer from prohibitive complexity. Importantly, all these approaches modify the internal architecture or message-passing scheme of GNNs, increasing model complexity. None of them treats structural knowledge as an explicit and decomposable representation that precisely extracts dataset-specific anomalous structures with linear computational overhead.

2.2 Graph Structural Anomaly Detection

Graph anomaly detection aims to identify nodes, edges, or subgraphs whose behavior or structure deviates significantly from the norm [29]. Early methods relied on handcrafted features (e.g., node degree, clustering coefficient) and traditional machine learning models such as isolation forests and one-class SVMs [30]. In recent years, the advent of Graph Neural Networks has brought new advances to graph anomaly detection.

GNN-based anomaly detection methods typically employ graph autoencoders or contrastive learning frameworks to learn node representations, identifying anomalies through reconstruction errors or representation discrepancies. DOMINANT [31] is the first work to apply GNNs to attributed network anomaly detection, using a graph convolutional autoencoder to reconstruct node attributes and graph structure, with reconstruction errors serving as anomaly scores. CoLA [32] adopts a contrastive learning framework that detects anomalies by maximizing mutual information between each node and its local neighboring substructure. DARGAT-GDN [33] combines graph attention networks with hyperbolic space embeddings to achieve robust performance in few-shot anomaly detection scenarios. SL-GAD [34] employs generative attribute reconstruction and multi-view contrastive learning to exploit contextual subgraph information, distinguishing anomalous nodes in both attribute and structure spaces. For dynamic graphs, motif-level anomaly detection methods [35] identify abnormal behaviors through the evolution patterns of temporal motifs. DPGAD [36] proposes a structure-aware dual-path attention network that explicitly models node roles in different motifs. Consequently, many of these methods do not explicitly model global high-order patterns, and those that do either rely on predefined motifs or lack task-specific discriminability.

Beyond the general graph anomaly detection approaches discussed above, researchers have also developed specialized models for specific application scenarios. For bot detection in social networks, BotRGCN [37] employs relational graph convolutional networks to model multiple relation types (e.g., follow, retweet, mention) between users, effectively capturing heterogeneous behavioral patterns of bots. These methods achieve strong performance on their respective tasks but are typically tailored to specific data characteristics (e.g., relation types), lacking generalizability. Moreover, they also rely on predefined relation types and cannot automatically discover unknown high-order anomalous structures.

2.3 Graph Neural Network Robustness and Adversarial Attacks

Another line of research relevant to our work focuses on GNN robustness against structural perturbations. Adversarial attacks on GNNs manipulate graph topology (e.g., adding or removing edges) to degrade node classification performance. Nettack [38], one of the earliest and most influential targeted attacks, greedily selects adversarial edge perturbations based on a linearized surrogate model, demonstrating that even slight structural modifications can cause dramatic performance degradation. From an optimization perspective, Xu et al. [39] formulated both targeted and global topology attacks as constrained optimization problems from a min-max perspective, exposing the vulnerability rooted in GNNs' reliance on local topological smoothness. Subsequently, MetaAttack [40] leveraged meta-learning to perform targeted structural perturbations, further showing that GNNs are fragile under adaptive, dataset-level adversarial manipulations.

In response, several robust GNNs have been proposed. RobustGCN [41] employs a Gaussian-based attention mechanism and propagates variance to absorb the impact of perturbed edges. Pro-GNN [42] jointly learns a clean graph structure and robust node representations. Beyond variance propagation and graph purification, another line of defense exploits the spectral properties of clean graphs. Adversarial edge perturbations typically introduce high-rank noise into the graph spectrum; Entezari et al. [43] proposed preprocessing the adjacency matrix via low-rank approximation (e.g., truncated SVD) to effectively restore the clean structural backbone and mitigate attack impact without modifying the GNN architecture. This spectral perspective complements the aforementioned feature-space and structure-learning defenses by targeting the inherent low-rank manifold of real-world networks. These robust GNNs are designed exclusively for general node classification under attack. Nevertheless, these dedicated robust GNNs typically achieve stronger defense than our incidental robustness, at the cost of architectural modifications.

Our SparseGNN takes a different approach: we do not introduce any dedicated robustness mechanism (variance propagation, adversarial training, or graph purification). Instead, we show that the proposed structural descriptors confer robustness by learning global atomic patterns, mitigating the impact of structural attack algorithms (e.g., Meta-Attack [40]).

In contrast, our proposed SparseGNN adopts a fundamentally different philosophy: it does not alter the backbone GNN architecture. Instead, it learns a sparse set of atomic subgraph patterns from the global graph in a label-balanced method, then constructs a structural descriptor that can be concatenated with original node features. This design keeps the base GNN unchanged, introduces linear computational overhead, and is plug-and-play for any standard GNN models. Consequently, we do not perform direct empirical comparisons with higher-order GNNs—their architectural assumptions and scalability profiles are orthogonal to our setting. A theoretical complexity analysis is provided in Section 3.6 to justify our design choices.

3 Method

3.1 Method Overview

Standard GNNs struggle to perceive global high-order topologies, and existing augmentation methods rely on predefined patterns. To address this issue, we propose the Sparse Atomic Structure Enhanced Graph Neural Network (SparseGNN). The core idea is to automatically learn a set of basic subgraph patterns (referred to as atomic subgraphs) from the global graph, and represent each node's local neighborhood structure as a sparse composition (Boolean union) of these atoms. Based on this representation, we compute for each node a continuous weight vector, where each entry indicates the contribution of the corresponding atom to reconstructing the node's local subgraph. This vector is then L2-normalized to obtain

a structural descriptor. If an atomic pattern is highly correlated with anomalous behavior, a high value in the corresponding dimension of the descriptor reveals the node's potential anomaly risk. Finally, the descriptor is concatenated with the original node features and fed into any standard GNN, enabling the model to perceive high-order anomalous topologies that would otherwise be hidden from local aggregation.

As illustrated in Fig. 1, SparseGNN consists of four stages. First, balanced sampling is performed on training nodes, local subgraphs are extracted and aligned, and sparse dictionary learning learns an initial atom dictionary $\mathbf{D}$. Second, using the fixed dictionary $\mathbf{D}$, sparse coding is solved for each node while recording the reduction in reconstruction error contributed by each selected atom, yielding a continuous weight vector $\mathbf{w}_v$. Third, the weight vector $\mathbf{w}_v$ is L2-normalized (or set to zero if it is a zero vector) to obtain the structural descriptor $\mathbf{s}_v$. Fourth, the descriptor $\mathbf{s}_v$ is concatenated with the original node features $\mathbf{x}_v$ and fed into any standard GNN for end-to-end training.

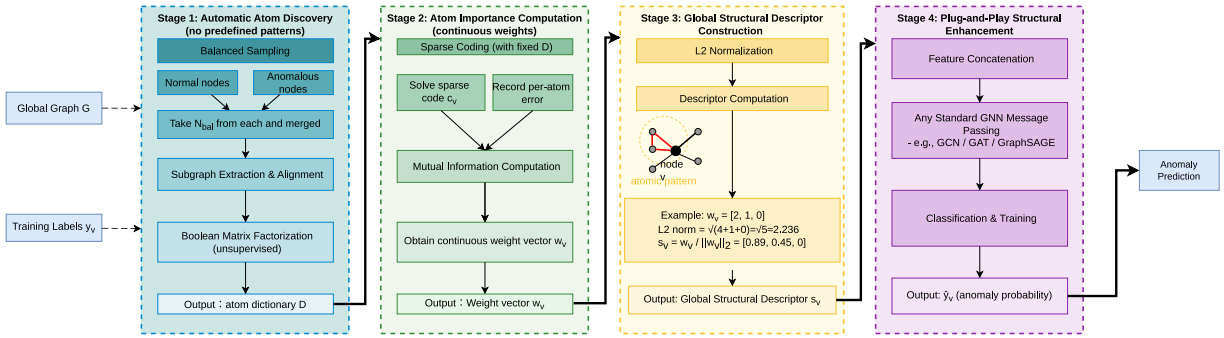

Figure 1: Overview of the proposed SparseGNN framework.

3.2 Atomic Structure Learning with Label-Balanced Sampling

To automatically learn a set of atomic subgraph patterns from the global graph that can represent both normal and anomalous high-order structures, we face a key challenge: if the input data is dominated by normal nodes, the learned dictionary will inevitably be biased toward normal structures, thereby overlooking rare but critical anomalous patterns. To address this issue, we introduce an atomic structure learning framework with label-balanced sampling. Specifically, training labels are used only to construct a class-balanced node subset, ensuring equal representation of normal and anomalous structures in the input data. The subsequent atom extraction process—including dictionary initialization, sparse coding, and dictionary update—is performed without accessing any label information. This design forces the factorization to cover both types of topological structures while avoiding dependence on predefined patterns or handcrafted features.

3.2.1 Balanced Sampling and Subgraph Alignment

Given the training node set V_{train} and their labels $y_v \in \{0,1\}$ (0 = normal, 1 = anomalous), let $N_{\text{norm}} = |\{v \in V_{\text{train}} : y_v = 0\}|$ and $N_{\text{anom}} = |\{v \in V_{\text{train}} : y_v = 1\}|$. We set the balanced sampling size as $N_{\text{bal}} = \min(N_{\text{norm}}, N_{\text{anom}})$. For instances, in anomalous users detection tasks, we randomly sample N_{bal} nodes without replacement from the normal class and N_{bal} from the anomalous class, respectively, and merge them to obtain the balanced subset V_{bal}.

The adoption of balanced sampling is motivated by two considerations. First, if we directly used all training nodes (where normal nodes typically outnumber anomalous ones by a large margin), the objective of sparse dictionary learning would be dominated by the reconstruction error of normal subgraphs. This

would bias the learned dictionary toward normal structures and make it difficult to capture rare but critical anomalous patterns. Balanced sampling forces the decomposition to treat both classes equally, thereby enabling the dictionary to cover both normal and anomalous high-order topological characteristics. Second, for social networks with millions of nodes, full-scale decomposition is infeasible by itself—this is precisely one of the problems that our balanced sampling addresses. However, if the number of nodes is not very large, full decomposition can be considered as an alternative. Balanced sampling dramatically cuts computational overhead while preserving dictionary representativeness.

For each node $v \in V_{\text{bal}}$, our goal is to sample a set of S nodes around it (including the center node itself) to obtain a fixed-size local subgraph representation. A straightforward method is to extract its h-hop ego-network (typically $h = 2$), sort the neighbors in descending order of their degree in the full graph G, and keep the center node along with its $S - 1$ highest-degree neighbors; alternatively, other sampling strategies can be adopted depending on the task. To obtain a unified vectorized representation for all subgraphs, we fix the center node at index 0 and assign indices $1, 2, \ldots$ to the remaining nodes according to the sampling order. If the actual number of sampled nodes is less than S, we pad with virtual nodes (represented as all-zero rows/columns in the adjacency matrix) to reach size $S \times S$; if it exceeds S, we retain only the first S nodes (the center node plus $S - 1$ sampled neighbors). The adoption of degree-based sorting is motivated by a structure-information-maximization principle under the fixed subgraph budget S. High-degree neighbors participate in more topological interactions and thus carry richer local structural signals. From an anomaly-detection perspective, anomalous users (e.g., bots and coordinated spreaders) often strategically connect to high-degree hubs to amplify their influence, making such neighbors discriminatively valuable. Moreover, high-degree nodes exhibit higher topological saliency across the graph, which reduces the geometric variance of aligned subgraph vectors and improves the stability of dictionary learning. While degree sorting is a greedy heuristic, it is widely adopted in subgraph sampling literature.

The aligned adjacency matrix is flattened into a column vector $\mathbf{y}_v \in \{0,1\}^{S^2}$. Concatenating all column vectors yields the input matrix $\mathbf{Y} \in \{0,1\}^{m \times n}$, where $m = S^2$ and $n = |V_{\text{bal}}|$. Padding nodes are represented as all-zero rows and columns in the adjacency matrix. When such a padded position is selected as part of an atom, it contributes no structural information; in practice, the greedy algorithm rarely selects all-zero rows because they do not reduce reconstruction error.

In the first stage, training labels are used solely for class-balanced sampling, which also makes the approach scalable to large graphs. This is a label-balanced design choice that ensures the learned dictionary captures both normal and rare anomalous patterns. The sparse dictionary learning objective itself does not access any label information. We emphasize that no test labels or test nodes are used in any stage.

The choice of balanced sampling size $N_{\text{bal}} = \min(N_{\text{norm}}, N_{\text{anom}})$ ensures that the minority class, typically anomalous users, is fully retained in the training subset. Because anomalous accounts are far less prevalent than normal ones in real-world social networks, N_{bal} equals the total number of anomalous nodes, meaning no anomalous node is discarded. Consequently, spatially clustered anomalous patterns (e.g., tightly coupled bot networks or coordinated misinformation groups) are preserved in their entirety, provided they are captured within the local subgraphs of the sampled nodes.

Nevertheless, we acknowledge that fixed-size balanced sampling incurs a representational trade-off: only a subset of normal nodes is retained. This may omit certain spatially clustered structures among normal users (e.g., dense hobbyist communities or tightly knit friend circles). From the perspective of anomaly detection, this omission is arguably acceptable, because our primary objective is to prevent the dictionary from being biased toward dominant normal structures. By deliberately under-sampling normal nodes, we ensure that anomalous topologies receive equal optimization pressure during dictionary learning.

In practice, N_{bal} should be chosen by jointly considering the total number of anomalous nodes, available memory budget, and the desired coverage of normal diversity. When the graph is small enough (e.g., several thousands of nodes) and class imbalance is moderate, one may alternatively perform full-graph decomposition using all training nodes.

3.2.2 Sparse Dictionary Learning for Atomic Subgraph Patterns

We formulate atom learning as a sparse dictionary learning problem. Given the subgraph vector matrix $\mathbf{Y} \in \mathbb{R}^{m \times n}$ constructed from the balanced node subset, we seek a dictionary $\mathbf{D} \in \{0,1\}^{m \times k}$ and a sparse code matrix $\mathbf{C} \in \mathbb{R}^{k \times n}$ such that $\mathbf{Y} \approx \mathbf{DC}$, where each column of $\mathbf{C}$ contains at most t nonzero entries. The objective is to minimize the reconstruction error subject to a sparsity constraint:

$$\min_{\mathbf{D},\mathbf{C}} \|\mathbf{Y} - \mathbf{DC}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{c}_j\|_0 \le t, \ \forall j = 1, \ldots, n. \tag{1}$$

We solve Problem (1) using the KSVD algorithm [44], which iteratively alternates between sparse coding and dictionary update. We formulate atom learning as a sparse dictionary learning problem [45]. The sparse coding step identifies, for each node, a small set of atoms that collaboratively minimize its local reconstruction residual. During this iterative optimization, let $\mathbf{r}_v$ denote the current residual vector for node v, initialized as $\mathbf{r}_v = \mathbf{y}_v$ (the flattened subgraph vector defined in Section 3.2.1). Whenever an atom $\mathbf{d}_i$ is activated in the sparse representation of node v (i.e., $C_{iv} \neq 0$), we quantify the reduction in squared L2 reconstruction error attributable to this atom as:

$$\Delta_{v,i} = \|\mathbf{r}_v \odot \mathbf{d}_i\|_2^2 = \sum_{j=1}^{m} (r_{v,j} \cdot d_{j,i})^2, \tag{2}$$

where $\odot$ denotes the Hadamard (element-wise) product. Intuitively, $\Delta_{v,i}$ equals the squared L2 norm of the overlap between the current residual and atom $\mathbf{d}_i$; for binary residuals and atoms, this is numerically equal to the number of 1-valued positions covered. We then explicitly update the weight vector and the residual via:

$$w_{v,i} \leftarrow w_{v,i} + \Delta_{v,i}, \quad \mathbf{r}_v \leftarrow \mathbf{r}_v \odot (\mathbf{1} - \mathbf{d}_i). \tag{3}$$

This augmentation does not change the asymptotic time complexity of KSVD; it only requires maintaining an additional $n \times k$ weight matrix alongside the residual. After convergence, we output the dictionary $\mathbf{D}$ and the weight vectors $\mathbf{w}_v$ for each node in the balanced subset.

It is worth noting that balanced sampling is primarily introduced to reduce computational overhead on large graphs and to prevent the dictionary from being biased toward the majority class. However, when the entire graph is relatively small (e.g., the number of nodes is around tens of thousands so that the adjacency matrix can be fully loaded into memory) and the class distribution does not exhibit severe imbalance (e.g., the minority class accounts for at least 10% of the training nodes), one may instead perform full-graph sampling, i.e., use all training nodes for atom learning. In this case, the input matrix $\mathbf{Y}$ for sparse dictionary learning is constructed from the subgraph vectors of all training nodes, without the need for a class-balanced subset. Full-graph sampling preserves the complete data distribution and may yield slightly better dictionary coverage in some scenarios. In practice, one can flexibly choose between balanced sampling and full-graph sampling depending on the graph size and class distribution; both options are seamlessly supported in our framework.

3.3 Atom Importance Computation

After obtaining the atom dictionary $\mathbf{D} \in \{0,1\}^{m\times k}$, we need to quantify the importance of each atom for reconstructing the local subgraph of every node v, resulting in a continuous weight vector $\mathbf{w}_v \in \mathbb{R}^k$, where $\mathbf{w}_v[i]$ denotes the cumulative squared L2 reduction in reconstruction error contributed by atom i when reconstructing the subgraph of node v. For nodes in the balanced subset V_{bal} used during atom learning, their weight vectors have already been obtained and saved by the modified sparse dictionary learning algorithm; thus no recomputation is required. For all other nodes (i.e., training nodes not in V_{bal} as well as all test nodes), we compute the weight vectors independently using the following greedy matching pursuit algorithm.

For a node v that requires independent computation, we first extract its local subgraph and flatten it into a vector $\mathbf{y}_v \in \{0,1\}^m$ following the procedure in Section 3.2.1. We initialize the residual vector $\mathbf{r} = \mathbf{f}_v$, the weight vector $\mathbf{w}_v = \mathbf{0}$, and the set of atoms $S = \varnothing$. We then repeat the following steps at most t times (where t is the upper limit of sparsity) or until the residual becomes zero. For each unselected atom $i \notin S$, we compute the reduction in squared L2 reconstruction error that would result from adding atom i to the current reconstruction. This reduction equals $\|\mathbf{r} \odot \mathbf{D}_{:,i}\|_2^2$, i.e., the squared magnitude of the overlap between the residual and the atom; for binary vectors, this is numerically equal to the number of overlapping 1-valued positions. We select the atom i^* that maximizes this reduction, then increase $\mathbf{w}_v[i^*]$ by that reduction amount. Next, we update the residual by setting to 0 all positions that are covered by the selected atom (i.e., where $\mathbf{D}_{:,i^*}$ has a 1). Finally, we add i^* to S. After the iteration, the non-zero entries of $\mathbf{w}_v$ correspond to the atoms, and their values represent the cumulative squared L2 reduction in reconstruction error contributed by each atom. If all reductions are zero (e.g., for an isolated node), we set $\mathbf{w}_v = 0$.

Since we retain all k atoms without any filtering, the completeness of the dictionary is preserved, and the subsequent GNN can automatically learn the relevance of each atom to the task. For nodes that require independent computation, the time complexity per node is $O(k \cdot m \cdot t)$, where $m = S^2$, and k and t are constants; thus the single-node cost is $O(1)$. Because nodes in the balanced subset V_{bal} avoid redundant computation, the overall complexity scales linearly with the total number of nodes $|\mathcal{V}|$ (i.e., $O(|\mathcal{V}|)$), encompassing both the training and test sets.

3.4 Structural Descriptor Construction

After obtaining the atom importance weight vector $\mathbf{w}_v \in \mathbb{R}^k$ for each node v, we convert it into a structural descriptor $\mathbf{s}_v$ to be injected into the graph neural network. Since the scale of $\mathbf{w}_v$ may vary significantly across nodes due to differences in local subgraph sizes, we apply L2 normalization to eliminate this scale discrepancy and make the descriptor reflect the relative distribution of atom importance:

$$\mathbf{s}_v = \begin{cases} \dfrac{\mathbf{w}_v}{\|\mathbf{w}_v\|_2}, & \text{if } \|\mathbf{w}_v\|_2 > 0, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{4}$$

Here $\|\cdot\|_2$ denotes the Euclidean norm. For nodes whose weight vector is zero (e.g., isolated nodes or nodes whose local subgraph cannot be reconstructed by any atom), the descriptor is set to the zero vector. Since all entries of $\mathbf{w}_v$ are non-negative, the normalized $\mathbf{s}_v$ has entries in $[0,1]$ and preserves the original sparsity pattern (zero entries remain zero). This descriptor intuitively represents the distribution of importance of different atomic patterns in the local structure of node v.

Computing the descriptor for each node takes $O(k)$ time, leading to a total complexity of $O(|N| \cdot k)$, which is linear in the number of nodes.

3.5 Structure-Aware Message Passing and Classification

After obtaining the structural descriptor $\mathbf{s}_v$, we concatenate it with the original node features $\mathbf{x}_v$ to form the initial representation of node v:

$$\mathbf{h}_v^{(0)} = \mathrm{CONCAT}(\mathbf{x}_v, \mathbf{s}_v) \in \mathbb{R}^{d_x+k}. \tag{5}$$

The set $\{\mathbf{h}_v^{(0)}\}_{v\in V}$ is then fed into any standard graph neural network (e.g., GCN, GAT, or GraphSAGE). The GNN propagates node representations through L layers of message passing, and the final representation $\mathbf{h}_v^{(L)}$ is passed to a linear classifier to output the anomaly probability $\hat{y}_v$ (for binary classification) or class probabilities (for multi-class tasks). The model is trained end-to-end by minimizing the cross-entropy loss:

$$\mathcal{L} = -\frac{1}{|V_{\text{train}}|} \sum_{v\in V_{\text{train}}} \left[y_v \log \hat{y}_v + (1-y_v) \log(1-\hat{y}_v) \right]. \tag{6}$$

with the multi-class cross-entropy used for multi-class tasks. During training, the dictionary $\mathbf{D}$ and the computation of weight vectors $\mathbf{w}_v$ are kept fixed; only the GNN parameters are updated. Since $k \ll d_x$ (typically $k = 20$ while d_x is the original feature dimension), the additional computational overhead introduced by concatenation is negligible.

3.6 Complexity Analysis

We analyze the time complexity of each stage in SparseGNN and compare it with existing structure-aware methods. Let N be the number of nodes, $|E|$ the number of edges, and $d_{\text{avg}} = 2|E|/N$ the average degree. The hidden dimension of the GNN is d and the number of layers is L. In our experiments, the constants are: aligned subgraph size S, total number of atoms $k = 20$, sparsity upper bound $t = 5$, and balanced sampling size $N_{\text{bal}} \ll N$.

Atom Learning (Section 3.2). We extract local subgraphs from the balanced subset ($2N_{\text{bal}}$ nodes) and perform sparse dictionary learning. The greedy algorithm has complexity $O(m \cdot 2N_{\text{bal}} \cdot k)$, where $m = S^2$. Since m, $|V_{\text{bal}}|$, and k are independent of the full graph size N (in label-balanced settings with limited labels), the atom learning stage has complexity $O(m \cdot |V_{\text{bal}}| \cdot k)$, which is constant with respect to N but with a moderate constant factor (e.g., $m = 900$, $k = 20$, $|V_{\text{bal}}| \sim 10^3$).

Atom importance computation (Section 3.3). For nodes in the balanced subset V_{bal}, the weight vectors are already obtained during atom learning and require no additional computation. For all other nodes (i.e., training nodes not in V_{bal} as well as all test nodes), each node requires a greedy matching pursuit with complexity $O(k \cdot m \cdot t)$. Since k, m, and t are constants, the cost per node is $O(1)$. Hence, the total complexity of this stage is $O(|V|)$, i.e., linear in the number of nodes. Although the constant factor is large, it is acceptable on modern hardware for graphs with millions of nodes.

Structural Descriptor Construction (Section 3.4). For each node, we compute the L2 norm and perform normalization, which takes $O(k)$ time per node. The total complexity is $O(|V| \cdot k) = O(|V|)$, linear in the number of nodes.

GNN Message Passing and Classification (Section 3.5). A standard GNN (e.g., GCN, GAT) has complexity $O(L \cdot |E| \cdot d^2)$. Concatenating the structural descriptor (dimension k) with the original node features (dimension d_x) increases the input dimension to $d_x + k$. Thus the GNN complexity becomes $O(L \cdot |E| \cdot (d_x + k)^2)$. Since $k \ll d_x$, this is approximately $O(L \cdot |E| \cdot d_x^2)$, which is the same order as that of a standard GNN.

Overall Complexity. The total complexity of SparseGNN is $O(|V|) + O(L \cdot |E| \cdot d_x^2)$, where the linear term comes from atom importance computation and descriptor construction, and the GNN part dominates.

In contrast, higher-order GNNs (e.g., k-GNN) have complexity $O(N^k)$ or $O(N^k \cdot d^2)$ in the worst case, making them infeasible for large graphs even for $k = 3$. Nested GNN (NGNN) requires encoding a local subgraph per node, leading to $O(N \cdot S^2 \cdot d^2)$, where the subgraph size S can be large in dense regions. Graph Substructure Network (GSN) is theoretically linear $O(|E| \cdot M \cdot \text{pattern\_size})$ but relies on handcrafted patterns and has a large constant factor. Therefore, SparseGNN achieves linear complexity comparable to standard GNNs while significantly improving the ability to perceive high-order anomalous structures.

4 Experiments

In this section, we conduct comprehensive experiments to evaluate the effectiveness and robustness of the proposed SparseGNN framework. The experiments cover two representative tasks: anomalous user detection in social networks and vulnerable node identification under graph adversarial attacks. Furthermore, we conduct ablation studies to validate the necessity of core components including balanced sampling, data-driven atom learning, and the structural descriptor. The following subsections detail the experimental setup, main results, and ablation analysis.

4.1 Experimental Setup

Datasets. We conduct experiments on two representative tasks: anomalous user detection in social networks (Task 1) and transductive node classification under graph adversarial attacks (Task 2), as summarized in Table 1. The dataset statistics are summarized in Table 1. For Task 1 (Bot Detection), we use Twibot-20 [46], a large-scale Twitter bot detection dataset containing user features, follow relationships, and tweet interaction graphs with annotations for real users and bot accounts, as well as FairGAD [47], a real-world benchmark dataset for fair graph anomaly detection. FairGAD comprises two globally connected social network datasets (Twitter and Reddit), provides binary anomaly labels (misinformation spreaders vs. normal users).

Table 1: Dataset statistics used in the experiments.

Dataset	Task Type	Nodes	Edges	Features	Classes
Twibot-20	Bot Detection	229,580	227,979	7680	2
FairGAD-Reddit	Bot Detection	9892	1,211,748	780	2
FairGAD-Twitter	Bot Detection	47,712	468,697	780	2
Cora	Node Classification	2708	5429	1433	7
PubMed	Node Classification	19,717	88,651	500	3
Citeseer	Node Classification	3327	4732	3703	6

Although FairGAD was originally designed for unsupervised anomaly detection, we adapt it to semi-supervised node classification because the dataset provides complete ground-truth labels, and this setting better reflects real-world platforms that possess a small amount of manually verified labels. The FairGAD-Reddit dataset contains 9892 nodes with an anomaly ratio of 13.68%, while FairGAD-Twitter contains 47,712 nodes with an anomaly ratio of 6.7%. To ensure consistent evaluation, we adopt a reproducible 60/20/20 random split for training/validation/test, and we stratify the split by both the anomaly label and the sensitive attribute, so that each split maintains the same class proportion (anomaly vs. normal) as the original dataset. Training labels are used solely for class-balanced sampling during atom learning; validation labels are used only for early stopping; test labels remain completely hidden until final evaluation.

For Task 2 (Transductive node classification), we select three citation network datasets: Cora, PubMed, and Citeseer.

To verify whether the structural knowledge enhancement incidentally improves robustness against structural perturbations—without introducing any dedicated defense mechanism—we extend the evaluation to a secondary task: vulnerable node identification under graph adversarial attacks. Specifically, we adopt three standard citation network datasets—Cora, PubMed, and Citeseer [48]—which are standard benchmarks for transductive node classification. Note that these datasets contain no anomalous users; instead, we perform multi-class node classification under Metattack to assess whether global atomic patterns mitigate the impact of local edge perturbations.

In Cora, nodes represent papers, edges denote citation relationships, and node categories correspond to paper topics; adversarial versions are generated by injecting perturbed edges (e.g., via Metattack) to evaluate the model's ability to identify vulnerable nodes surrounded by perturbed edges. PubMed is a biomedical literature citation network with a structure similar to Cora and is also perturbed for adversarial evaluation. Citeseer serves as the third scientific paper citation network benchmark. All datasets adopt the official splits.

Baseline Methods

We select two categories of baseline models that are common to both tasks: standard graph neural networks and structure-augmented graph neural networks. The standard GNN architectures adopted as backbones include Graph Convolutional Network (GCN) [17], Graph Attention Network (GAT) [18] that learns node representations via multi-head self-attention, Graph Isomorphism Network (GIN) [49] which generalizes the Weisfeiler-Lehman test to maximize discriminative power, and the inductive model GraphSAGE [19]. These models rely solely on local neighbor message passing and lack explicit awareness of high-order structural patterns. The structure-augmented GNNs include Graph Substructure Network (GSN) [26] and Nested Graph Neural Network (NGNN) [27]. GSN injects structural information via predefined subgraph isomorphism counts (e.g., triangles, stars), while NGNN enhances expressiveness by extracting and independently encoding a local subgraph for each node. These baselines are used to evaluate the improvement in structural awareness achieved by our SparseGNN.

In addition, we introduce task-specific baselines. For Task 1 (bot detection), we adopt BotRGCN [37] as a dedicated baseline. BotRGCN is a bot detection model based on relational graph convolutional networks. In our experiments, we carefully tune both its neural architecture (e.g., number of layers, relation fusion strategy, hidden dimensions) and data preprocessing (feature selection, normalization, graph construction) to achieve its best performance on the three bot detection datasets, making it a task-customized strong baseline. For Task 2 (vulnerable node identification under graph adversarial attacks), we adopt RobustGCN [41] as a robustness baseline.

RobustGCN [41] is a graph adversarial robustness method that represents each node as a Gaussian distribution and propagates variance to defend against edge perturbations. It is primarily designed for general node classification on citation networks, not for anomalous user detection in social networks. Here we adopt it as a robustness baseline to compare the performance degradation under attacks. We emphasize that our SparseGNN incorporates no specialized robustness mechanisms (e.g., variance propagation or adversarial training); it only enhances structural awareness via the proposed descriptor. It should be emphasized that our SparseGNN does not incorporate any specialized robustness mechanisms (e.g., variance propagation or adversarial training); it only enhances structural awareness via the proposed descriptor. Therefore, the comparison with RobustGCN aims to demonstrate the difference between specialized robustness-enhanced models and our approach, rather than claiming SparseGNN as a superior robust model.

Evaluation Metrics

For both tasks, we adopt Accuracy and F1-score as the evaluation metrics. All metrics are computed on the test set, and we report the mean ± standard deviation over multiple runs.

Implementation Details

All models adopt a unified training configuration: Adam optimizer with an initial learning rate of 0.01, weight decay of 5e−4, and dropout rate of 0.5. The number of GNN layers is set to 2, and the hidden dimension is set to 128. For the proposed SparseGNN framework, the number of atoms $k = 20$, the subgraph sampling scale $S = 30$. The sparsity upper bound in sparse coding is set to $t = 5$. The detailed configuration of adversarial attacks follows established practices. For Metattack (global attack), we perturb 5% of the total edges using a greedy strategy to maximize classification error. All attacks are implemented using publicly available code with default parameters. To discover atomic subgraph patterns that cover both normal and anomalous structures, we use labels only for balanced sampling; the sparse dictionary learning process itself does not require label information.

4.2 Main Experimental Results and Analysis

In this section, we evaluate the proposed SparseGNN framework on bot detection and graph adversarial attack tasks, comparing it with various baseline methods.

4.2.1 Bot Detection Task

Table 2 reports the accuracy and F1-score of all methods on the Twibot-20 and FairGAD datasets. It is worth noting that the original FairGAD benchmark was designed for unsupervised anomaly detection, where models rely solely on reconstruction errors or contrastive learning without accessing any labels. We adapt it to a semi-supervised node classification setting. This adaptation is better aligned with real-world deployment scenarios, where social platforms typically possess a small amount of high-quality manually verified labels. Specifically, we utilize the complete global graph structure during training transductive setting while only hiding the labels of test nodes. This allows us to evaluate the detection performance more objectively under practical conditions.

Table 2: Classification performance (%) on bot detection datasets (Accuracy/F1-score). Best results are highlighted in bold. Underlined values indicate the second best results. Results are reported as mean ± standard deviation over 10 independent runs with different random seeds.

Method	Twibot-20	FairGAD-Reddit	FairGAD-Twitter
GCN	76.14 ± 0.89/79.19 ± 0.82	70.09 ± 0.92/41.27 ± 0.98	88.97 ± 0.87/51.45 ± 1.15
GAT	80.23 ± 0.62/81.45 ± 0.48	64.58 ± 0.58/37.91 ± 0.67	87.95 ± 0.88/49.52 ± 0.73
GIN	77.89 ± 0.65/72.86 ± 0.78	60.53 ± 0.72/35.15 ± 0.85	84.88 ± 0.92/42.72 ± 0.88
GraphSAGE	82.56 ± 0.44/83.68 ± 0.65	79.43 ± 0.53/54.82 ± 0.77	91.26 ± 0.82/57.41 ± 0.75
GSN	83.90 ± 0.62/84.02 ± 0.58	78.03 ± 0.68/55.35 ± 0.65	89.56 ± 0.80/53.51 ± 0.85
NGNN	83.37 ± 0.65/83.77 ± 0.62	77.09 ± 0.72/53.61 ± 0.70	90.21 ± 0.82/52.04 ± 0.84
BotRGCN	**85.09 ± 0.65/85.76 ± 0.57**	<u>80.27 ± 0.72/58.12 ± 0.68</u>	<u>91.53 ± 0.85/58.15 ± 0.90</u>
SparseGCN (Ours)	80.37 ± 0.58/81.68 ± 0.67	75.82 ± 0.65/53.32 ± 0.75	90.58 ± 0.82/52.45 ± 0.78
SparseSAGE (Ours)	<u>84.12 ± 0.49/84.21 ± 0.55</u>	**81.53 ± 0.56/58.38 ± 0.62**	**92.43 ± 0.72/59.52 ± 0.68**

To demonstrate the plug-and-play capability of our framework, we inject the structural knowledge descriptor into two representative GNN backbones—GCN and GraphSAGE—yielding SparseGCN and SparseSAGE, respectively. Their classification performance is reported in Table 2 alongside other baselines. The following observations can be made.

First, except for GraphSAGE, standard GNN methods all perform poorly. On the FairGAD-Reddit dataset, the F1-scores of GCN, GAT, and GIN are only 41.27%, 37.91%, and 35.15%, respectively, indicating

that these models relying solely on local neighbor aggregation struggle to capture anomalous patterns hidden in complex higher-order topologies. Notably, the anomaly label ratio in this dataset is only 13.68%, indicating a severe class imbalance. Under such imbalance, transductive or attention-based models such as GCN, GAT, and GIN tend to bias toward the majority class (normal nodes), leading to a significant drop in their ability to identify the minority class (anomalous nodes). In contrast, GraphSAGE achieves a much higher F1-score of 54.82%, significantly outperforming other standard GNNs. This can be attributed to GraphSAGE's inductive learning capability and neighbor sampling mechanism, which allows GraphSAGE to encounter more diverse local subgraph structures during training, making it more robust to class imbalance.

Second, structure-augmented GNNs (GSN, NGNN) show consistent improvement over standard GNNs, but still lag significantly behind our method. For instance, on FairGAD-Reddit, GSN achieves 55.35% on F1-score and NGNN achieves 53.61%, while our SparseSAGE reaches 58.38%; on FairGAD-Twitter, GSN and NGNN obtain 53.51% and 52.04% F1-score, respectively, while SparseSAGE achieves 59.52%. This demonstrates the advantage of data-driven atomic structures over predefined subgraph patterns (e.g., triangles, stars).

Third, our proposed SparseGNN achieves the best results in most cases. Specifically, on FairGAD-Reddit and FairGAD-Twitter, SparseSAGE outperforms all competing methods, with accuracy/F1 of 81.53%/58.38% and 92.43%/59.52%, respectively. On Twibot-20, SparseSAGE ranks second (84.12%/84.21%), only behind the carefully tuned BotRGCN (85.09%/85.76%), and better than other models such as GSN (83.90%/84.02%).

Fourth, our method (SparseGNN) is a plug-and-play enhancement framework. When inserted into GCN and GraphSAGE, it yields significant improvements in classification accuracy across multiple datasets. Specifically, SparseGCN outperforms vanilla GCN by approximately 5.73% (Accuracy) on FairGAD-Reddit, and SparseSAGE outperforms vanilla GraphSAGE by approximately 2.10% (Accuracy) on FairGAD-Reddit. This demonstrates that equipping graph neural networks with additional structural awareness is generally beneficial for discriminating higher-order anomalous topologies. Notably, the improvement margin on GraphSAGE is relatively smaller. This is not because structural enhancement is ineffective, but rather because GraphSAGE, through its neighbor sampling and inductive aggregation mechanism, already possesses a certain degree of inherent structural awareness (as analyzed earlier). Consequently, the marginal gain of injecting atomic structural descriptors is smaller on top of an already structure-aware model. In contrast, for GCN, which has weaker structural awareness, the enhancement brought by our method is much more pronounced. These results further confirm the effectiveness and adaptability of SparseGNN as a general-purpose augmentation module.

Fifth, the superior performance of BotRGCN on Twibot-20 can be explained by the data characteristics. Twibot-20 is not a single globally connected graph; instead, it consists of multiple disjoint second-order ego-networks centered at seed users. Each subgraph is relatively isolated, lacking global cross-subgraph higher-order interactions. In such a local island data structure, our method's reliance on global atomic structure decomposition cannot fully exploit its ability to capture cross-region high-order patterns. In contrast, BotRGCN is specifically optimized for the relation types (e.g., follow, retweet, mention) and graph structure of this dataset, including tailored relational graph convolution and feature selection. Nevertheless, without any dataset-specific tuning, SparseGNN still outperforms all models except BotRGCN, confirming its strong generalization as a generic augmentation framework.

Sixth, the generally low F1-scores on the FairGAD datasets (the highest is around 59.52% on SparseSAGE, FairGAD-Twitter) reflect inherent challenges. Anomalous users (misinformation spreaders) and users with sensitive attributes (e.g.,right-leaning users) exhibit highly similar behavioral patterns, causing models to easily misclassify right-leaning users as anomalies. This makes binary classification extremely difficult. Nonetheless, our method (SparseSAGE) achieves 58.38% and 59.52% F1 on Reddit and Twitter,

respectively, outperforming BotRGCN (58.12% and 58.15%) and all other baselines. This indicates that SparseGNN possesses a certain capability to distinguish between these two highly similar user groups, mitigating the confusion caused by sensitive attributes to some extent.

4.2.2 Accuracy Retentio under Graph Adversarial Attacks

The goal of this experiment is not anomaly detection, but to verify whether SparseGNN can improve classification robustness against local edge perturbations by enhancing the model's perception of global high-order topology via structural descriptors. To this end, we perform multi-class semi-supervised node classification on three standard citation networks (Cora, PubMed, Citeseer) and apply Meta-Attack [40] to globally perturb the graph structure. It is worth noting that these datasets contain no anomalous nodes and are multi-class. To adapt the experiment, we extend the binary class balanced sampling to multi-class sampling, i.e., sample the same number of nodes from each class, ensuring the atom dictionary covers topological patterns of all classes.

Except for the above adaptations, the atom importance computation, structural descriptor construction, and GNN integration of SparseGNN remain unchanged. This preserves the generality of our conclusions and avoids introducing any dedicated robustness mechanisms (e.g., variance propagation, adversarial training). Specifically, we apply the Meta-Self variant (the best-performing global attack variant in the original work) with a perturbation ratio of 5% (i.e., modifying 5% of the edges in the largest connected component (LCC) of each graph) on the Cora, PubMed, and Citeseer datasets. Table 3 reports classification accuracy on clean and attacked graphs, with the drop indicated by the ↓ symbol. We focus primarily on accuracy drop, which directly reflects the model's sensitivity to structural perturbations.

Table 3: Classification accuracy changes of models under adversarial attack on citation network datasets. The values in the table are presented as Classification Accuracy Rate Before Attack (BA%)/Classification Accuracy After Attack (AA%) ↓ Classification Accuracy Drop (AD%), where the accuracy drop is calculated by subtracting the accuracy after attack from the accuracy before attack. Best results are highlighted in bold. Underlined values indicate the second best results.

Method	Cora	PubMed	Citeseer
	BA/AA ↓AD	**BA/AA ↓AD**	**BA/AA ↓AD**
GCN	83.6/76.1 ↓7.5	71.8/70.5 ↓1.3	<u>87.5/83.6 ↓3.9</u>
GAT	**83.5/80.8 ↓2.7**	73.7/72.2 ↓1.5	83.2/78.7 ↓4.5
RobustGCN	83.1/77.8 ↓5.3	**71.4/70.9 ↓0.5**	86.2/81.1 ↓5.1
SparseGCN (Ours)	<u>86.1/81.0 ↓5.1</u>	<u>73.5/72.3 ↓1.2</u>	**89.7/87.6 ↓2.5**

The following observations can be made. First, all methods suffer from varying degrees of performance degradation under adversarial attacks, indicating that structural perturbations of the graph have a universal impact on node classification tasks.

Second, the proposed SparseGNN method achieves competitive robustness across all datasets. Specifically, it attains the best performance on the Citeseer dataset (2.5%) and the second-best performance on both Cora and PubMed (5.1% and 1.2%). Meanwhile, compared with the vanilla GCN, SparseGCN achieves higher classification accuracy on clean graphs across all datasets (e.g., 2.5% on Cora). This demonstrates that our method enhances the model's intrinsic representation capability while simultaneously improving its resistance to structural disturbances (5% edge perturbations), thereby achieving a synergistic improvement in both accuracy and robustness.

Third, it should be emphasized that our method is not specifically designed for robustness. Its resistance to perturbations stems from the atomic structural descriptors that encode global higher-order topology, rather than from any dedicated robustness mechanisms (e.g., variance propagation, adversarial training). Consequently, when the intensity of structural perturbations increases to a level that destabilizes the global atomic representations of the entire graph (e.g., excessively high perturbation ratios or attack strategies that systematically distort global patterns), the structural awareness foundation of our method will be undermined, leading to a significant drop or even complete loss of the enhancement effect.

In summary, experiment demonstrates that SparseGNN can simultaneously enhance the accuracy and robustness of clean graphs under low-intensity structural perturbation.

4.3 Ablation Study

To systematically validate the effectiveness of each core component in SparseGNN, we conduct ablation experiments using GCN as the backbone network. We compare the following five model variants on two representative tasks: bot detection (FairGAD-Twitter) and node classification under adversarial attacks (Cora-Metattack with 5% edge perturbation).

- GCN (Baseline): Standard graph convolutional network without any structural augmentation.
- GCN + Random Atoms: Replaces the learned atom dictionary with a randomly initialized fixed set of atoms (same number of atoms $k = 20$), while keeping the rest of the pipeline unchanged.
- GCN + Predefined Motifs: Uses a deliberately minimal set of four handcrafted topological primitives—triangle, 3-node path, 4-node star, and their isomorphic variants—as the structural features.
- GCN + Sparse Descriptor (Full): The complete proposed model with balanced sampling, data-driven atom learning, structural descriptor construction, and GCN backbone.

All variants adopt the same training configuration as described in Section 4.2. Table 4 reports results on the two datasets.

Table 4: Ablation study results on FairGAD-Twitter (Accuracy %) and Cora under Meta-attack (Classification Accuracy Rate Before Attack (BA%)/Classification Accuracy After Attack (AA%) ↓ Classification Accuracy Drop (AD%)). Best results are highlighted in bold.

Method	FairGAD-Twitter	Cora-Metattack
	Accuracy (%)	BA/AA ↓AD
GCN (Baseline)	88.97	83.6/76.1 ↓7.5
GCN + Random Atoms	86.75	82.1/75.2 ↓6.9
GCN + Predefined Motifs	89.24	85.1/79.1 ↓6.0
GCN + Sparse Descriptor (Full)	**92.43**	**86.1/81.0 ↓5.1**

From Table 4, we observe that random atoms are not beneficial; GCN + Random Atoms performs even slightly worse than the baseline GCN, indicating that random patterns cannot capture real graph topology and instead introduce noise. In contrast, GCN + Predefined Motifs improves over the baseline, but its gain is substantially smaller than that of the full model, demonstrating that data-driven atom learning discovers richer and more task-relevant structural patterns. Furthermore, the full model (SparseGCN) achieves the best performance on all metrics, significantly surpassing all ablated variants. This validates the necessity of each designed component: balanced sampling ensures coverage of anomalous patterns, data-driven atom

learning discovers high-value structures, and the structural descriptor injects global topological information into node representations.

The large performance gap between GCN + Predefined Motifs and the full model is not merely a matter of motif quantity. SparseGNN's advantage stems from its ability to discover dataset-specific anomalous structures that lie outside human preset imagination—for instance, the non-standard community shapes and cross-layer coupling patterns exhibited by misinformation spreaders in FairGAD, which no fixed motif inventory can capture. Predefined motifs rely on isomorphism counting, producing a scalar count per motif for every node; these features are static, unweighted, and cannot be adaptively combined. In contrast, SparseGNN's structural descriptor is a continuous weight vector obtained via sparse linear combination of atoms, whose expressive capacity grows linearly with k not only in dimensionality but also in combinatorial possibility, because each node can select a different subset of atoms. Consequently, expanding the predefined set to the scale of GSN (7–10 patterns) would yield marginal gains.

In summary, the simplified predefined baseline in our ablation precisely proves the central claim of this paper: data-driven atom discovery outperforms predefined patterns not because it employs more patterns, but because it discovers more accurate patterns by automatically identifying the structural primitives most discriminative from global topology for the current task. Any fixed inventory is inherently unable to cover the structural diversity of unknown datasets.

5 Conclusion

This paper proposes SparseGNN, a structural-knowledge-enhanced framework for anomalous user detection in social networks. Its core innovation lies in learning discriminative atomic subgraph patterns from global topology via data-driven sparse coding, and injecting them as compact structural descriptors into any standard GNN. By preserving the backbone architecture and incurring only linear computational overhead, SparseGNN significantly improves the perception of high-order anomalous topologies without the prohibitive cost of higher-order GNNs. Extensive experiments on bot detection and adversarial robustness tasks demonstrate that SparseGNN serves as an effective, general-purpose enhancement framework. Ablation studies further validate the necessity of data-driven atom learning and structural descriptors over random or handcrafted patterns. Notably, the observed robustness against adversarial structural perturbations is an incidental byproduct of encoding global topology, rather than the result of any dedicated defense mechanism.

In practice, SparseGNN is best suited for large-scale social-network platforms that already possess a small amount of labeled anomalous accounts and seek a plug-and-play enhancement to standard GNN backbones without architectural overhaul. Nevertheless, because the atom dictionary is learned from the specific topological distribution of each individual network, its generalization across heterogeneous social platforms with divergent structural characteristics remains to be investigated.

Acknowledgement: None.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Zehan Li and Yingyi Li; methodology, Zehan Li, Yingyi Li and Zhiwei Tang; software, Zehan Li; validation, Zehan Li, Yingyi Li and Xuemeng Zhai; formal analysis, Zehan Li; investigation, Zehan Li and Jiandong Liang; resources, Yingyi Li and Guangmin Hu; data curation, Zehan Li and Xuemeng Zhai; writing—original draft preparation, Zehan Li; writing—review and editing, Yingyi Li, Zhiwei Tang, Xuemeng Zhai, Jiandong Liang and Guangmin Hu; visualization, Zehan

Li; supervision, Yingyi Li and Guangmin Hu; project administration, Yingyi Li. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Data openly available in a public repository.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ferrara E, Varol O, Davis C, Menczer F, Flammini A. The rise of social bots. Commun ACM. 2016;59(7):96–104. doi:10.1145/2818717.
2. Zhang Y. Social network user profiling for anomaly detection based on graph neural networks. In: Proceedings of the 2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA); 2025 Mar 28–30; Xi'an, China. New York, NY, USA: IEEE; 2025. p. 1197–201.
3. Jethani V, Pathak V. Spam and fake account detection using machine learning: a review. Int J Recent Res Rev. 2025;18(1):230–40.
4. Pacheco D, Hui PM, Torres-Lugo C, Truong BT, Flammini A, Menczer F. Uncovering coordinated networks on social media: methods and case studies. In: Proceedings of the International AAAI Conference on Web and Social Media (ICWSM); 2021 Jun 7–10; Virtual. p. 455–66.
5. Stringhini G, Kruegel C, Vigna G. Detecting spammers on social networks. In: Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC); 2010 Dec 6–10; Austin, TX, USA. New York, NY, USA: ACM; 2010. p. 1–9.
6. Sahoo SR, Gupta BB, Peraković D, Peñalvo FJG, Cvitić I. Spammer detection approaches in online social network (OSNs): a survey. In: Sustainable management of manufacturing systems in Industry 4.0. Berlin/Heidelberg, Germany: Springer; 2022. p. 159–80.
7. Lee EW, Bao H, Wang Y, Lim YT. From pandemic to Plandemic: examining the amplification and attenuation of COVID-19 misinformation on social media. Soc Sci Med. 2023;328(5):115979. doi:10.1016/j.socscimed.2023.115979.
8. Boshmaf Y, Muslukhov I, Beznosov K, Ripeanu M. The socialbot network: when bots socialize for fame and money. In: Proceedings of the 27th Annual Computer Security Applications Conference (ACSAC); 2011 Dec 5–9; Orlando, FL, USA. New York, NY, USA: ACM; 2011. p. 93–102.
9. Xing L, Li S, Zhang Q, Wu H, Ma H, Zhang X. A survey on social network's anomalous behavior detection. Complex Intell Syst. 2024;10(4):5917–32. doi:10.1007/s40747-024-01446-8.
10. Hooi B, Song YA, Beutel A, Shah N, Shin K, Faloutsos C. FRAUDAR: bounding graph fraud in the face of camouflage. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2016 Aug 13–17; San Francisco, CA, USA. p. 895–904.
11. Jiang M, Cui P, Beutel A, Faloutsos C, Yang S. CatchSync: catching synchronized behavior in large directed graphs. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2014 Aug 24–27; New York, NY, USA. p. 941–50.
12. Ghosh S, Viswanath B, Kooti F, Sharma NK, Korlam G, Benevenuto F, et al. Understanding and combating link farming in the twitter social network. In: Proceedings of the 21st International Conference on World Wide Web; 2012 Apr 16–20; Lyon, France. p. 61–70.
13. Abu-El-Haija S, Perozzi B, Kapoor A, Alipourfard N, Lerman K, Harutyunyan H, et al. MixHop: higher-order graph convolutional architectures via sparsified neighborhood mixing. In: Proceedings of the 36th International Conference on Machine Learning (ICML); 2019 Jun 10–15; Long Beach, CA, USA. p. 21–9.
14. Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS. A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst. 2020;32(1):4–24. doi:10.1109/tnnls.2020.2978386.
15. Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering. Adv Neural Inf Process Syst. 2016;29:3844–52.
16. Veličković P. Everything is connected: graph neural networks. Curr Opin Struct Biol. 2023;79:102538.

17. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. In: Proceedings of the International Conference on Learning Representations (ICLR); 2017 Apr 24–26; Toulon, France.
18. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph attention networks. In: Proceedings of the International Conference on Learning Representations (ICLR); 2018 Apr 30–May 3; Vancouver, BC, Canada.
19. Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. Adv Neural Inf Process Syst. 2017;30:1025–35.
20. Alon U, Yahav E. On the bottleneck of graph neural networks and its practical implications. In: Proceedings of the International Conference on Learning Representations (ICLR); 2021 May 3–7; Virtual.
21. Xue G, Zhong M, Qian T, Li J. PSA-GNN: an augmented GNN framework with priori subgraph knowledge. Neural Netw. 2024;173:106155. doi:10.1016/j.neunet.2024.106155.
22. Morris C, Ritzert M, Fey M, Hamilton WL, Lenssen JE, Rattan G, et al. Weisfeiler and leman go neural: higher-order graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2019 Jan 27–Feb 1; Honolulu, HI, USA. Vol. 33, p. 4602–9.
23. Maron H, Ben-Hamu H, Shamir N, Lipman Y. Invariant and equivariant graph networks. arXiv:1812.09902. 2018.
24. Zhang S, Hu Z, Subramonian A, Sun Y. Motif-driven contrastive learning of graph representations. IEEE Trans Knowl Data Eng. 2024;36(8):4063–75. doi:10.1109/tkde.2024.3364059.
25. Shervashidze N, Schweitzer P, Van Leeuwen EJ, Mehlhorn K, Borgwardt KM. Weisfeiler-lehman graph kernels. J Mach Learn Res. 2011;12(9):2539–61.
26. Bouritsas G, Frasca F, Zafeiriou S, Bronstein MM. Improving graph neural network expressivity via subgraph isomorphism counting. IEEE Trans Pattern Anal Mach Intell. 2022;45(1):657–68. doi:10.1109/tpami.2022.3154319.
27. Zhang M, Li P. Nested graph neural networks. In: Proceedings of the 35th International Conference on Neural Information Processing Systems; 2021 Dec 6–14; Virtual. Red Hook, NY, USA: Curran Associates Inc.; 2021. p. 15734–47.
28. Chen Z, Villar S, Chen L, Bruna J. On the equivalence between graph isomorphism testing and function approximation with GNNs. Adv Neural Inf Process Syst. 2019;32:15894–902.
29. Ma X, Wu J, Xue S, Yang J, Zhou C, Sheng QZ, et al. A comprehensive survey on graph anomaly detection with deep learning. IEEE Trans Knowl Data Eng. 2021;35(12):12012–38. doi:10.1109/tkde.2021.3118815.
30. Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Comput Surv. 2009;41(3):1–58.
31. Ding K, Li J, Bhanushali R, Liu H. Deep anomaly detection on attributed networks. In: Proceedings of the 2019 SIAM international Conference on Data Mining; 2019 May 2–4; Calgary, AB, Canada. Philadelphia, PA, USA: SIAM; 2019. p. 594–602.
32. Liu Y, Li Z, Pan S, Gong C, Zhou C, Karypis G. Anomaly detection on attributed networks via contrastive self-supervised learning. IEEE Trans Neural Netw Learn Syst. 2021;33(6):2378–92. doi:10.1109/tnnls.2021.3068344.
33. Yang S. DARGAT-GDN: a robust and adaptive graph neural network for few-shot graph anomaly detection. In: Proceedings of the 2025 5th International Symposium on Computer Technology and Information Science (ISCTIS); 2025 May 16–18; Xi'an, China. New York, NY, USA: IEEE; 2025. p. 190–9.
34. Zheng Y, Jin M, Liu Y, Chi L, Phan KT, Chen YPP. Generative and contrastive self-supervised learning for graph anomaly detection. IEEE Trans Knowl Data Eng. 2021;35(12):12220–33. doi:10.1109/tkde.2021.3119326.
35. Huang C, Yu B, Gao C, Tu Y, Jiang F, Huang X. Structural-temporal mining for motif-level anomaly detection in dynamic graphs. Knowl-Based Syst. 2025;325(5):113962. doi:10.1016/j.knosys.2025.113962.
36. Dong X, Zhang H, Han H, Xu Z. DPGAD: structure-aware dual-path attention graph node anomaly detection. Symmetry. 2025;17(9):1452. doi:10.3390/sym17091452.
37. Feng S, Wan H, Wang N, Luo M. BotRGCN: twitter bot detection with relational graph convolutional networks. In: Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining; 2021 Nov 8–11; The Hague, The Netherlands. p. 236–9.
38. Zügner D, Akbarnejad A, Günnemann S. Adversarial attacks on neural networks for graph data. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; 2018 Aug 19–23; London, UK. p. 2847–56.

39. Xu K, Chen H, Liu S, Chen PY, Weng TW, Hong M, et al. Topology attack and defense for graph neural networks: an optimization perspective. arXiv:1906.04214. 2019.
40. Zügner D, Günnemann S. Adversarial attacks on graph neural networks via meta learning. In: Proceedings of the International Conference on Learning Representations (ICLR); 2019 May 6–9; New Orleans, LA, USA.
41. Zhu D, Zhang Z, Cui P, Zhu W. Robust graph convolutional networks against adversarial attacks. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; 2019 Aug 4–8; Anchorage, AK, USA. p. 1399–407.
42. Jin W, Ma Y, Liu X, Tang X, Wang S, Tang J. Graph structure learning for robust graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; 2020 Aug 23–27; San Diego, CA, USA. p. 66–74.
43. Entezari N, Al-Sayouri SA, Darvishzadeh A, Papalexakis EE. All you need is low (rank) defending against adversarial attacks on graphs. In: Proceedings of the 13th International Conference on Web Search and Data Mining; 2020 Feb 3–7; Houston, TX, USA. p. 169–77.
44. Aharon M, Elad M, Bruckstein A. K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans Signal Process. 2006;54(11):4311–22.
45. Zhai X, Zhou W, Fei G, Lu C, Hu G. Network sparse representation: decomposition, dimensionality-reduction and reconstruction. Inf Sci. 2020;521(2):307–25. doi:10.1016/j.ins.2020.02.022.
46. Feng S, Wan H, Wang N, Li J, Luo M. TwiBot-20: a comprehensive twitter bot detection benchmark. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management; 2021 Nov 1–5; Gold Coast, QSL, Australia.
47. Neo NKN, Lee YC, Jin Y, Kim SW, Kumar S. Towards fair graph anomaly detection: problem, benchmark datasets, and evaluation. In: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management; 2024 Oct 21–25; Boise, ID, USA. New York, NY, USA: ACM. p. 1752–62.
48. Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T. Collective classification in network data. AI Mag. 2008;29(3):93–3. doi:10.1609/aimag.v29i3.2157.
49. Xu K, Hu W, Leskovec J, Jegelka S. How powerful are graph neural networks? In: Proceedings of the International Conference on Learning Representations; 2019 May 6–9; New Orleans, LA, USA.