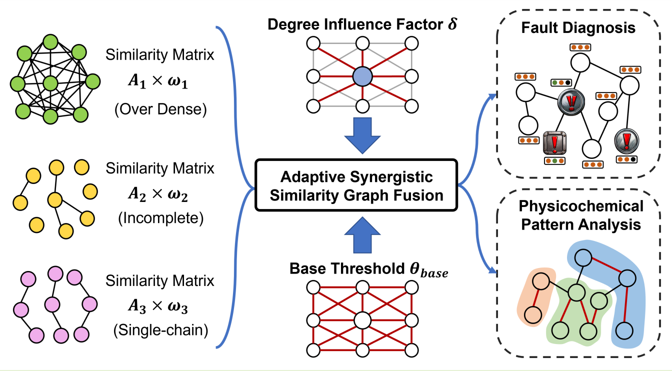
A-SSGC: Adaptive Graph Construction Capturing Physicochemical Commonalities for Industrial Fault Diagnosis

Yifan Chen, Haiqi Zhu, Zhiyuan Chen, Haoxuan Xu, Dario Landa-Silva, and Hafeez Ullah Amin, *Senior Member, IEEE*

***Abstract*—Accurate identification of subtle faults in industrial manufacturing remains a critical challenge, driving increased adoption of machine learning (ML) techniques. However, classical ML models often overlook complex inter-sample relationships rooted in shared physicochemical properties, thereby compromising diagnostic accuracy. Addressing this, we propose Adaptive Synergistic Similarity Graph Construction (A-SSGC), a novel algorithm that adaptively fuses multiple graph construction methods. A-SSGC employs an adaptive sparsification strategy, guided by node degrees, to capture physicochemical commonalities among samples effectively. A-SSGC significantly outperforms traditional ML models, basic graph construction techniques, and both unsupervised and semi-supervised deep graph construction approaches. It consistently outperforms these baselines across representative graph neural networks on multiple industrial manufacturing datasets. Visualization of the constructed graphs confirms the ability of A-SSGC to reveal physicochemical commonalities, thereby enhancing interpretability and supporting deeper analytical insights. By effectively capturing these commonalities, A-SSGC improves diagnostic performance. It also shows strong potential as a versatile tool for industrial data analysis, contributing to improved automation and reliability in manufacturing processes. Our code and datasets are available at `https://github.com/AnguoCYF/A-SSGC`.**



***Index Terms*—Graph Construction Methods, Graph Neural Networks, Industrial Fault Diagnosis, Physicochemical Commonalities**

Author's Accepted Manuscript. Released under the Creative Commons license: Attribution 4.0 International (CC BY 4.0)
https://creativecommons.org/licenses/by/4.0/

I. Introduction

INDUSTRIAL materials, such as steel plates and glass, are essential to numerous sectors including construction, transportation, and energy. They play a critical role in supporting infrastructure, vehicle manufacturing and machinery production, significantly impacting both daily life and the global economy [1]–[3]. Given their widespread application, ensuring the quality and reliability of these materials is paramount. Consequently, fault diagnosis in industrial materials has become a crucial area of research, influencing quality control, operational efficiency, and end-user safety [4], [5].

Despite its importance, fault diagnosis faces considerable challenges due to the complexity and scale of data obtained from industrial production environments. Traditional analytical methods are generally suitable for capturing linear and obvious defect patterns. However, they often fail to detect more subtle and nonlinear patterns. These challenging patterns typically result from intricate chemical compositions, complex processing parameters, and diverse microstructures [6], [7]. To address these limitations, machine learning (ML) techniques have gained popularity. Classical ML methods such as Support Vector Machines (SVM), Multilayer Perceptron (MLP), Decision Trees, and Random Forest (RF) have been extensively studied and applied to industrial fault diagnosis, delivering promising results [8]–[11].

This work was supported in part by the National Key Research and Development Program under Grant 2024YFC3016400 and in part by the Ministry of Industry and Information Technology of China.

Corresponding authors: Haiqi Zhu and Zhiyuan Chen.

Y. Chen and Z. Chen are with the School of Computer Science, University of Nottingham Malaysia, Semenyih, Selangor, 43500, Malaysia (e-mail: hcxyc1@nottingham.edu.my; Zhiyuan.Chen@nottingham.edu.my).

H. Zhu is with the Zhengzhou Research Institute, Harbin Institute of Technology, Zhengzhou 450000, Henan, China, AND also with the School of Medicine and Health, Harbin Institute of Technology, Harbin, Heilongjiang, 15001, China (e-mail: haiqizhu@hit.edu.cn).

H. Xu is with the Faculty of Computing, Harbin Institute of Technology, Harbin, Heilongjiang, 15001, China (e-mail: 23s136156@stu.hit.edu.cn).

H. U. Amin is with the Department of Computer Science, Edge Hill University, Ormskirk, Lancashire, L39 4QP, United Kingdom (e-mail: hafeezullah.amin@edgehill.ac.uk).

D. Landa-Silva is with the School of Computer Science, University of Nottingham, Nottingham, NG7 2RD, United Kingdom (e-mail: dario.landasilva@nottingham.ac.uk).

While these traditional ML methods offer promising results, they typically treat samples as independent entities, neglecting the inherent physicochemical relationships among them. This assumption limits their diagnostic effectiveness, as correlated defects arising from shared material properties, processing parameters, or environmental conditions remain unaccounted for [12]–[15]. Graph-based learning, which effectively leverages relationships among samples, has gained significant attention across numerous domains [16]–[19]. However, its application in industrial manufacturing remains challenging due to the absence of explicit relational data.

Developing effective graph construction techniques to capture these intrinsic material attributes is therefore essential. However, as far as we know, dedicated research efforts focused on generalizable graph construction strategies remain limited. In most research, graph construction methods are treated as auxiliary components tied to the development of specific graph neural network (GNN) models, resulting in limited adaptability beyond narrow applications.

Classical approaches often rely on simplistic heuristics, such as applying a fixed threshold to a single pairwise similarity metric, failing to adaptively integrate diverse relationship types. Recent deep graph construction methods have explored sophisticated approaches-including embedding-driven adjacency generation, probabilistic graph decoding through latent variable models, and iterative topology optimization [20]–[23].

However, these methods typically initiate the process from a single similarity perspective or learn a unified latent embedding space, neglecting the multi-dimensional nature of sample relationships. Even when advanced methods incorporate adaptive or fusion-based principles, they often exhibit critical limitations. For instance, adaptive methods learn graph structures from node embeddings. They are designed as components tightly coupled with a specific GNN, limiting the model-agnostic utility [22]. Similarly, fusion-based graph structure learning approaches often operate in a supervised or semi-supervised manner, integrating graph fusion within the GNN model's training loop [24]. This fundamentally differs from our framework, which is fully unsupervised and model-agnostic. Furthermore, both approaches often lack an explicit mechanism to control graph density [20]–[22], [24]. Such implicit 'one-size-fits-all' assumptions can introduce evaluation biases, as the optimal connections among samples vary significantly across different industrial materials and processes [25]–[28].

To overcome these limitations, this paper proposes the Adaptive Synergistic Similarity Graph Construction (A-SSGC) framework. A-SSGC is a novel approach designed for industrial fault diagnosis. It adaptively integrates multiple graph construction strategies. Each strategy captures unique sample relationships based on different measurement principles. The framework also uses an adaptive sparsification strategy. This prevents overly dense local connections and constructs a sparse, comprehensive graph. This graph encapsulates the underlying material properties among samples. Finally, GNNs leverage the resulting structure to accurately detect faults.

The primary contributions of this research are summarized as follows:

1) **Enhanced Diagnostic Performance:** A-SSGC significantly improves fault diagnosis accuracy across multiple industrial manufacturing scenarios.
2) **Broadened Applicability of Graph-Based Learning:** A-SSGC captures physicochemical commonalities in diverse contexts, extending the applicability of graph-based learning methods to industrial scenarios lacking explicit relational data.
3) **Improved Interpretability and Reliability:** A-SSGC constructs sparse, semantically meaningful graphs that align with domain-specific physicochemical relationships, offering visually verifiable and auditable decision pathways for industrial operators.

This research builds upon and significantly extends our preliminary work on graph construction. This foundational study introduced the Synergistic Similarity Graph Construction (SSGC) framework, which demonstrated the initial promise of fusing multiple similarity metrics [29]. We introduce A-SSGC, featuring a novel adaptive sparsification mechanism based on node degrees for improved graph construction. Compared to the preliminary study focused only on steel plates, this paper provides substantially broader contributions: A-SSGC is validated across diverse industrial materials and benchmarked against numerous basic and deep graph construction methods. Furthermore, we present detailed graph visualization analyses and rigorous ablation and sensitivity studies, demonstrating A-SSGC's enhanced effectiveness, interpretability, and robustness. The remainder of this paper is structured as follows: Section II reviews related work on traditional ML and graph-based methods in fault diagnosis. Section III elaborates on the A-SSGC framework, detailing the foundational graph construction methods, adaptive sparsification strategy, employed GNNs, and evaluation metrics. Section IV presents experimental analyses, detailing datasets, preprocessing, and comparative performance of various methods. Finally, Section V concludes the study and outlines potential future directions.

II. Related Works

A. Traditional ML Models for Fault Diagnosis

Fault diagnosis is a key field of engineering technology. Its purpose is to identify and locate the fault points in the system or equipment. Fault diagnosis plays an important role in industrial fields such as energy, transportation, and manufacturing [4], [5]. Efficient fault diagnosis can ensure normal operation of equipment, mitigate risks, reduce downtime, and prolong service life [30], [31]. With the increasing complexity and scale of modern industrial systems, the requirements for fault diagnosis accuracy and efficiency are becoming higher and higher.

Traditional fault diagnosis methods are mainly based on physical models and empirical rules. For example, Moreira and Lesage [32] investigated fault diagnosis recognition based on discrete-event models. Zheng and Zhou [33] explored the training effect of principle knowledge on fault diagnosis performance. In addition, Hu et al. [34] proposed an extended qualitative multi-fault diagnosis method based on the principle knowledge. However, with the growth of system complexity and data size, these traditional methods encounter difficulties in dealing with large-scale and complex fault diagnosis tasks.

Fault diagnosis in industrial settings often involves complex data characterized by subtle but significant physicochemical relationships between different fault instances [35]–[37]. Traditional ML models typically operate under an assumption of i.i.d., ignoring inherent correlations among samples. This assumption can severely limit their diagnostic effectiveness in real-world scenarios, where failures are often interconnected due to shared material properties, manufacturing conditions, or environmental factors [38]–[40].

To meet this challenge, researchers have utilized ML techniques to improve fault diagnosis for industrial materials. Tian et al. [8] introduced the SVM model to detect steel plate faults. By employing recursive feature elimination and utilizing evolutionary optimization algorithms, they achieved an accuracy 80.7%, meeting the fault detection requirements. Nkonyana et al. [9] employed three ML models—RF, MLP, and SVM for steel plate fault classification. Their comparative analysis revealed that ensemble methods outperformed basic ML models in detecting faults within the steel manufacturing process. Inspired by these findings, Chen et al. [10] employed a series of ensemble strategies based on decision trees for steel plate fault classification and found the Bagging algorithm to be particularly effective. This algorithm outperformed other methods, achieving a test accuracy of 90%, which represents an effective method for detecting surface abnormalities on steel plates. Overall, these studies demonstrate the feasibility of traditional ML techniques for detecting industrial materials.

However, despite being relatively effective for fault diagnosis, conventional ML models have limitations due to they treat samples as isolated instances. This independence assumption neglects the physical commonalities shared among industrial materials. Such commonalities might arise from material characteristics, manufacturing processes, and the operational environment, which all can influence the appearance of faults [41], [42]. For example, variations in the chemical composition of materials or inconsistencies in the cooling process during manufacturing could lead to correlated faults like surface defects and internal stress fractures [43], [44]. These physical commonalities form a graph of interconnected fault patterns, where the presence suggests the potential for others. Traditional models fail to consider these interconnected patterns, compromising diagnostic performance [35], [36], [45].

B. Graph-based Learning for Fault Diagnosis

Graphs, comprising nodes and edges, provide a potent mathematical framework for representing and analyzing relationships between entities. Unlike data modalities such as time series or images, graphs can explicitly model complex dependencies and structural information [16]–[19], making them highly suitable for domains with intricate inter-sample relationships.

By utilizing the graph's capability to model inter-sample relationships, which are often overlooked by traditional methods, GNNs have emerged as a significant advancement in graph-based learning, offering robust mechanisms for capturing these structural dependencies. Notable GNN architectures include Graph Convolutional Networks (GCN) [46] for effective local structure aggregation, Graph Attention Networks (GAT) [47] for dynamically prioritizing influential neighbors, and Graph Isomorphism Networks (GIN) [48] for distinguishing nuanced graph-level patterns. These capabilities have been successfully demonstrated in diverse applications, such as GCN's effectiveness in molecular property prediction [49], GAT's superior performance in social network recommendations [50], and GIN's exceptional capacity to distinguish complex chemical structures [51].

However, the efficacy of GNNs heavily relies on the quality of the input graph, a critical challenge in industrial contexts that often lack explicit relational data. Dedicated research on effective and generalizable graph construction strategies remains limited. Furthermore, most graph construction development is treated as an auxiliary component tied to specific GNN models, restricting adaptability beyond narrow applications [22], [23].

Existing graph construction methods can be broadly categorized. Conventional methods often use fixed similarity metrics or simple heuristics. For instance, cosine similarity has been used for dynamic adjacency matrices in motor imagery classification [25], k-nearest neighbor (k-NN) strategies for modeling cell similarities in scRNA-seq data clustering [26], and modified Minimum Spanning Tree (MST) based graphs for data hierarchies [27]. Although some studies have compared multiple graph metrics in human activity recognition [28], these conventional approaches can be sensitive to the choice of metrics and parameters, struggling to capture the multifaceted inter-sample dependencies.

Beyond conventional learning approaches, recent advances in deep graph construction research have expanded into more complex and specialized scenarios. A prominent direction is hybrid deep graph construction, which seeks to integrate diverse information sources. This is particularly prevalent in biomedical applications, where strategies include building and integrating multiple patient similarity networks from multi-omics data [52]. To handle the inherent noise and complexity in such settings, self-supervised learning is increasingly employed to refine and denoise the constructed graph structures by creating supervisory signals from the data itself [53]. These principles converge in highly complex settings like federated learning, where recent work focuses on simultaneously learning both graph semantic and structural information across decentralized clients [54]. A common characteristic of these advanced methods is their focus on refining an existing graph or designing task-specific, end-to-end pipelines. They thus offer significant utility in specialized scenarios, but pay less attention to the foundational construction of a graph from data when no prior structure is available.

Deep learning-based graph construction methods have emerged, primarily following four technical routes:

1) **Embedding-based methods**, such as GraphVAE [21], learn to decode graph structures from latent variables. However, they often produce overly dense adjacency

matrices, introducing noise and reducing interpretability, which is problematic for high-dimensional, noisy industrial data.

2) **Latent structure refinement methods**, such as Learning Discrete Structures initialized by k-Nearest Neighbor (kNN-LDS) [20], iteratively refine graphs from an initial heuristic structure k-NN. These semi-supervised methods depend heavily on the quality of the initial graph and require labeled data, limiting performance when initial graphs are suboptimal or labeled data is scarce.
3) **Embedding-driven self-adaptive methods**, such as the graph learning component in MTGNN [22], dynamically generate adjacency matrices from learnable node embeddings. As auxiliary components coupled to specific GNNs, these methods often exhibit poor generalization and instability, particularly with sparse or noisy industrial data.
4) **Hybrid and Self-Supervised Graph Construction** which integrates multiple data sources with deep-learning modules, often utilizing self-supervision for refinement in complex settings such as in multi-modal biomedical applications and distributed federated learning systems. [52]–[54].

In summary, existing methods face critical limitations, including issues with graph density, dependence on initial graphs and labeled data, and restricted generalizability or structural instability. These shortcomings underscore the need for a novel graph construction method. This method should be able to adaptively combine multiple metrics and ensure sparse, interpretable structures. It should also operate without labeled data and remain robust to initialization and data noise. Such criteria guide the development of the proposed A-SSGC method.

III. Methodology

A. Concepts and Definitions

In this study, we model industrial fault diagnosis as a node classification task, where each node in the graph represents an industrial sample with associated features. The objective is to classify each sample node into one of several fault categories based on its features and relationships to other nodes. The relationships between nodes are represented by edges, which capture the physicochemical commonalities between samples, such as shared physical or chemical properties.

The adjacency matrix A is used to represent the connections between nodes. Each entry in the matrix, $\mathbf{A}_{ij}$, indicates the strength or existence of a connection between node i and node j. The feature matrix X contains the attributes of each node, with each row corresponding to a node's features, such as material properties relevant to fault diagnosis. The graph is constructed based on these matrices, allowing the model to leverage both node features and the captured inter-sample relationships for fault classification.

B. Basic Graph Construction Methods

In the field of industrial fault diagnosis, graph construction methods offer a powerful approach to leverage the relationships among samples. By representing samples as nodes and their relations as edges within a graph, these methods effectively capture the inherent structure of the data, thereby boosting the efficacy of fault detection.

K-Nearest Neighbors: This strategy forms the graph by associating each sample with its k nearest neighbors based on Euclidean distance. The adjacency matrix A for k-NN is defined as follows:

$$A_{ij} = \begin{cases} 1 & \text{if the } j_{th} \text{ sample is a } \mathrm{k-NN} \text{ of the } i_{th} \text{ sample} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

In Eq. (1), i_{th} and j_{th} represent the indices of the sample in the dataset. k-NN effectively captures local structures by considering the k closest data points for each sample, thereby enabling the detection of localized anomalies that indicate faults. However, one potential limitation of this method is that it may overlook global patterns, as it primarily focuses on local relations [55].

Cosine Similarity: This method utilizes cosine similarity to calculate the similarity between the i_{th} and the j_{th} samples using Eq. (2):

$$A_{ij} = \frac{x_i \cdot x_j}{|x_i|_2 \, |x_j|_2} \tag{2}$$

Here, x_i and x_j represent the feature vectors of the respective samples, and $|\mathrm{x_i}|_2$ and $|\mathrm{x_j}|_2$ denote the Euclidean norms of these vectors. The raw cosine similarity scores from Eq. (2), which range from [-1, 1], are typically processed for graph construction. In our approach, an adjacency matrix A_i is formed where $A_{ij} = 1$ if the cosine similarity between sample i and j surpasses a predefined threshold, and $A_{ij} = 0$ otherwise. This results in a binary matrix with values in $\{0, 1\}$.While this method efficiently identifies samples with similar directions, its lack of scale consideration may give rise to misleading associations [56].

Euclidean Distance: This measurement calculates the distance between two data points in Euclidean space, which represents the shortest path between them. It is widely used in vector space models due to its straightforward computation and effectiveness for dense or continuous data. Edges are created if the Euclidean distance falls below a predetermined threshold. The Euclidean distance between two samples is expressed as:

$$D_{ij} = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \tag{3}$$

In the Eq. (3), x_{ik} and x_{jk} denote the k-th feature of samples x_i and x_j and x_j respectively, and n indicates the total number of features. The adjacency matrix A_i is constructed where $A_{ij} = 1$ if the Euclidean distance D_{ij} falls below a predetermined threshold, and $A_{ij} = 0$ otherwise. This yields a binary matrix.

Mahalanobis Distance: This method accounts for feature correlations when measuring the distance between samples. The Mahalanobis distance between the i-th and j-th samples is calculated as (4):

$$D_{ij} = \sqrt{\left(x_i - x_j\right)^\top S^{-1} \left(x_i - x_j\right)} \tag{4}$$

Here, x_i and x_j are the feature vectors, and S^{-1} is the inverse of the covariance matrix S of the dataset. The adjacency matrix A_i is generated where $A_{ij} = 1$ if the Mahalanobis distance D_{ij} is below a predefined threshold, and $A_{ij} = 0$ otherwise. This generated a binary matrix with values in $\{0, 1\}$. This method effectively takes data scale and feature correlations into account. However, it depends on an accurate covariance matrix estimation. In cases of small sample sizes or multicollinearity, such an estimation may be unreliable [57].

Minimum Spanning Tree: The MST algorithm ensures connectivity among all data points by minimizing the sum of edge weights, which corresponds to the Euclidean distances between samples. The MST can be formally expressed as Eq. (5):

$$T^* = \underset{T\subseteq E}{\operatorname{argmin}} \sum_{(i,j)\in T} A_{ij} \tag{5}$$

T^* signifies the set of edges constituting the MST, while E represents all edges in the fully connected graph. A_{ij} is the Euclidean distance between sample i and sample j. The operator $\underset{T\subseteq E}{\operatorname{argmin}}$ denotes the operation of finding the tree T which minimizes the total weight of all the edges as computed by $\sum_{(i,j)\in T} A_{ij}$. The constraint that T forms a tree ensures that all data points are connected without forming any cycles. The corresponding adjacency matrix A_i is binary, where $A_{ij} = 1$ if an edge (i, j) is part of the MST, and $A_{ij} = 0$ otherwise. While this guarantees global connectivity, the MST technique might not fully capture local data structures. This is because the MST algorithm selects edges based on global connectivity criteria, often overlooking finer, localized relationships between data points. Consequently, it may fail to represent more intricate local structures, which are crucial for understanding the underlying patterns in the data [58].

Given the strengths and limitations of each method, a novel graph construction algorithm A-SSGC is proposed, to blend these methods and construct a robust and representative graph.

C. A-SSGC: Adaptive Synergistic Similarity Graph Construction

In this work, we present A-SSGC, as depicted in Figure 1. Initially, five well-established methods are employed to generate adjacency matrices: k-NN, Cosine Similarity, Euclidean Distance, Mahalanobis Distance and MST. These measurements provide a balanced representation of spatial, structural, and similarity-based relations, particularly suited for modeling the physical and chemical commonalities within industrial materials.

Each adjacency matrix captures unique inter-sample relations. The fusion strategy of A-SSGC is grounded in the principles of ensemble learning, where combining diverse models is known to yield more robust and stable results than any single constituent model [59]. Specifically, our approach is analogous to consensus clustering, where the objective is to find a superior consensus structure from multiple, varied inputs [60]. A-SSGC integrates these matrices by assigning different weights w_i, unifying them into a single graph structure that represents this consensus. Details on computing the weights w_i will be elaborated further in this section.

The primary advantage of the A-SSGC algorithm lies in its ability to integrate diverse information from different similarity matrices. A-SSGC operates under the assumption that adjacency matrices with similar structures are likely to contain key information, therefore should be given higher importance in the fusion process. Conversely, matrices with significant differences may contain less critical or noisy information and are assigned smaller weights. This strategy ensures the efficient fusion of adjacency matrices and generates a robust and representative graph structure for subsequent analysis. The specific steps of the A-SSGC method are detailed in Algorithm 1.

Algorithm 1 Adaptive Synergistic Similarity Graph Construction (A-SSGC)

```
Input: Set of adjacency matrices $\{A_i\}_{i=1}^n$ from various graph construction methods; Base threshold $\theta_{\text{base}}$; Degree factor $\delta$
Output: Fused adjacency matrix $A_{\text{fused}}$
for $i \leftarrow 1$ to $n$ do
    for $j \leftarrow 1$ to $n$ do
        $d_{ij} \leftarrow \|A_i - A_j\|_F$
        $s_{ij} \leftarrow e^{-d_{ij}}$
    end for
    $w_i \leftarrow \frac{\sum_{j=1}^n s_{ij}}{\sum_{i=1}^n \sum_{j=1}^n s_{ij}}$
    $A_{\text{fused}} \leftarrow \sum_{i=1}^n w_i \cdot A_i$
end for
for each node $u$ do
    $degree_u \leftarrow \sum_{v \neq u} I(A_{\text{fused}}[u, v] > 0)$    ▷ Compute node degrees
end for
for each node $u$ do
    $norm\_degree_u \leftarrow \frac{degree_u - d_{\min}}{d_{\max} - d_{\min}}$    ▷ Normalize node degrees
end for
for each pair $(u, v), u \neq v$ do
    $\theta_{uv} \leftarrow \theta_{\text{base}} + \delta \times \frac{norm\_degree_u + norm\_degree_v}{2}$    ▷ Compute adaptive thresholds
    $\theta_{uv} \leftarrow \min(1, \max(0, \theta_{uv}))$    ▷ Ensure $\theta_{uv} \in [0, 1]$
    if $A_{\text{fused}}[u, v] < \theta_{uv}$ then
        $A_{\text{fused}}[u, v] \leftarrow 0$    ▷ Remove the edge
    end if
end for
for each node $u$ do
    $A_{\text{fused}}[u, u] \leftarrow 1$    ▷ Add self-loops
end for
Return $A_{\text{fused}}$
```

Algorithm 1 begins by measuring the similarity between the adjacency matrices obtained from various graph construction methods, the Frobenius norm is employed to quantify the distance between these matrices. This distance is then transformed into a similarity measure. The computation of the

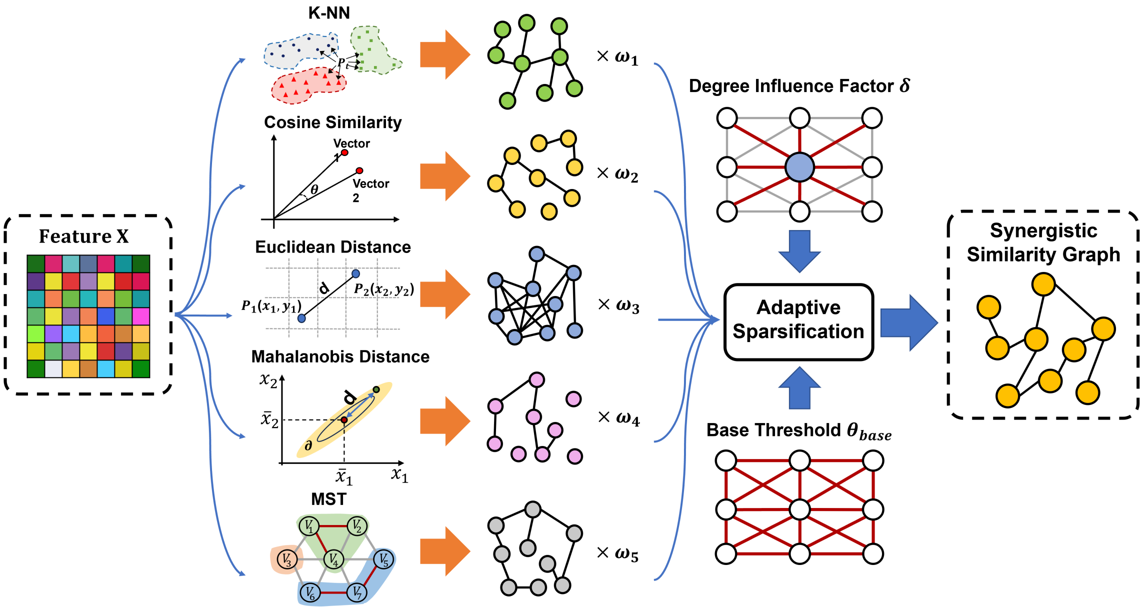


Fig. 1: Schematic Diagram of the A-SSGC Method

similarity is expressed as:

$$\text{similarity}\left(A_i, A_j\right) = e^{-|A_i - A_j|F} \tag{6}$$

In Eq. (6), the Frobenius norm $|\cdot|_{\mathrm{F}}$ is utilized to quantify the overall difference between the adjacency matrices A_i and A_j generated by different graph construction methods. To transform this difference into a meaningful similarity measure, the exponential function e^{-x} is applied to the Frobenius norm. This transformation enhances the discriminative power of the similarity metric. It assigns higher values to matrices that exhibit greater resemblance and, conversely, lower values to matrices that are more dissimilar.

Based on the similarity between the adjacency matrices, the weights w_i are calculated to reflect the importance of each adjacency matrix in the fusion process. To compute the weights, we normalize the similarities so that their sum equals 1 :

$$w_i = \frac{\sum_{j=1}^{n} \text{similarity}\left(A_i, A_j\right)}{\sum_{i=1}^{n} \sum_{j=1}^{n} \text{similarity}\left(A_i, A_j\right)} \tag{7}$$

In Eq. (7), the weight for each of the n adjacency matrices is calculated based on their similarity to the ensemble. Since the values in these input matrices range from 0 to 1, this weighting and normalization scheme ensures that matrices with more prominent connections have a proportionally greater impact on the final fused graph.

The theoretical rationale for this weighting scheme is to find a robust consensus structure from an ensemble of diverse graphs. This objective is formally conceptualized as finding the Fréchet mean of the graph ensemble. The Fréchet mean is defined as the geometric centroid A^* that minimizes the sum of squared distances to all graphs in the set [61]:

$$A^* = \arg\min_{A'} \sum_{i=1}^{n} ||A' - A_i||_F^2 \tag{8}$$

In this objective function, the set $\{A_1, ..., A_n\}$ represents the input adjacency matrices, and the goal is to find an ideal matrix A' that is maximally central to this set. While directly solving this optimization problem is non-trivial, the weighting scheme in Eq. (7) serves as a computationally efficient approximation. The weight w_i assigned to each matrix A_i quantifies its centrality within the ensemble, which serves as a proxy for its proximity to the geometric centroid A^*. Therefore, by amplifying the influence of more central matrices, the resulting weighted average is guided toward the optimal consensus solution.

This ensemble-based approach also provides strong stability. A core tenet of statistical learning is that combining diverse models can reduce variance and improve robustness over any single model [59]. The weighting formula in Eq. (6) possesses an inherent outlier suppression mechanism that contributes to this stability. If a specific construction method produces an outlier matrix A_i that is distant from all others, the exponential terms $e^{-||A_i - A_j||_F}$ will be close to zero due to the large distances. Consequently, its weight w_i will be minimal, preventing a single noisy graph from destabilizing the fusion process. The calculated weights are then used to perform a weighted average of all adjacency matrices, resulting in a fused adjacency matrix denoted as A_{fused}:

$$A_{\text{fused}} = \sum_{i=1}^{n} w_i \cdot A_i \tag{9}$$

In Eq. (9), A_{fused} is obtained by multiplying each adjacency matrix A_i with its corresponding weight w_i and summing the results.

To further optimize the fused adjacency matrix, an adaptive threshold strategy is introduced to binarize A_{fused}, enhancing its ability to capture meaningful relationships. A-SSGC calculates adaptive thresholds for each pair of nodes, starting by computing the degree of each node as Eq.(10):

$$\text{degree}_u = \sum_{v \neq u} \mathbb{I}\left(A_{\text{fused}}[u, v] > 0\right) \tag{10}$$

where $\mathbb{I}(\cdot)$ is the indicator function. which equals 1 if the condition is true, and 0 otherwise. The degrees are then normalized to range $[0, 1]$:

$$\text{norm\_degree}_u = \begin{cases} 0, & \text{if } d_{\max} = d_{\min} \\ \frac{\text{degree}_u - d_{\min}}{d_{\max} - d_{\min}}, & \text{otherwise} \end{cases} \tag{11}$$

In Eq.(11), $d_{\min}$ and $d_{\max}$ are the minimum and maximum degrees among all nodes, respectively. For each pair of nodes (u, v), the adaptive threshold θ_{uv} is computed as Eq.(12):

$$\theta_{uv} = \theta_{base} + \delta \times \frac{\text{norm\_degree}_u + \text{norm\_degree}_v}{2} \tag{12}$$

where θ_{base} is the base threshold, and δ is the degree factor. The threshold θ_{uv} is then clamped to the interval $[0, 1]$ to ensure valid values. The fused adjacency matrix A_{fused} is binarized by comparing each element with its corresponding adaptive threshold: If $A_{\text{fused}}[u, v] \geq \theta_{uv}$ the edge is retained. If $A_{\text{fused}}[u, v] < \theta_{uv}$ the edge is removed by setting $A_{\text{fused}}[u, v] = 0$. Finally, self-loops are added by setting $A_{\text{fused}}[u, u] = 1$ for all nodes u.

A-SSGC dynamically adjusts the threshold for each edge based on the normalized degrees of the connected nodes, preventing over-dense local connections and better capturing meaningful relationships. This approach results in a more representative and discriminative graph structure by harnessing the strengths of individual graph construction methods while mitigating their weaknesses. When combined with GNNs, A-SSGC provides a nuanced representation of sample interconnections, improving the industrial fault diagnosis performance and extending the applicability of graph-based learning to industrial scenarios lacking explicit relational data.

D. Graph Neural Networks

Once the graph is constructed, GNNs are utilized to identify underlying anomalous patterns through the graph data for fault diagnosis, as shown in Figure 2. Unlike traditional methods, this framework specifically tailors graph-based learning for industrial fault diagnosis. The innovative integration of 'Graph Construction' and 'Graph Neural Networks' modules enhances the diagnosis process by capturing and exploring the chemical and physical commonalities within industrial materials.

Figure 2 illustrates the workflow for industrial materials fault diagnosis. The process begins with 'Feature Preprocessing', where sample features are standardized to facilitate more effective subsequent analyses. This phase is followed by 'Graph Construction,' which translates the standardized data into a graph format: each node represents an individual sample and edges encapsulate the inherent relations. Building upon this graph structure, the analysis naturally progresses into the utilization of GNNs.

In this study, we select three representative GNN models: GCN, GAT, and GIN to validate the versatility and robustness of the proposed A-SSGC algorithm across various GNNs. Each model offers unique strengths for exploring the complex constructed graph, as detailed below:

Graph Convolutional Network: GCN is a seminal model in the GNNs that introduced a localized first-order approximation of spectral graph convolutions. In this model, each node aggregates information from its neighboring nodes, thus emphasizing homophily-based features. The standard convolution operation of GCN can be mathematically represented as:

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i) \cup i} \frac{1}{\sqrt{d_i d_j}} W^{(l)} h_j^{(l)}\right) \tag{13}$$

In Eq. (13), $h_i^{(l+1)}$ denotes the feature of node i at layer $l+1$. The normalization components, d_i and d_j, which represent node degrees, ensuring the stabilization of the model's training. $W^{(l)}$ signifies the weight matrix at layer l, and σ acts as the non-linear activation function.

Given its equal weight distribution mechanism, GCN offers robustness in detecting anomalies, capitalizing on local graph structures, and rendering it proficient in discerning underlying patterns within industrial materials.

Graph Attention Network: Among the selected GNNs, GAT stands out with its attention mechanisms. The fundamental operation of GAT is aggregating the features from neighboring nodes weighted by attention coefficients. The attention coefficients α_{ij} are calculated using a shared attentional mechanism, allowing the model to focus on informative neighbors that are more likely to share similar fault characteristics. This is mathematically represented as Eq. (14):

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} W^{(l)} h_j^{(l)}\right) \tag{14}$$

In Eq. (14), $h_i^{(l+1)}$ is the feature of node i at layer $l+1, \mathcal{N}(i)$ is the set of neighbors of node $i, W^{(l)}$ is the learnable weight matrix in layer $l, h_j^{(l)}$ is the feature of node j in layer l , and $\sigma(\cdot)$ is the activation function.

The attention coefficients α_{ij} are computed by Eq. (15):

$$\alpha_{ij} = \text{softmax}_i\left(\text{LeakyReLU}\left(a^T\left[W^{(l)} h_i^{(l)} \| W^{(l)} h_j^{(l)}\right]\right)\right) \tag{15}$$

Where $\|$ denotes concatenation, a is a learnable weight vector, and $\text{softmax}_i(\cdot)$ ensures the coefficients for neighbors of node i sum to 1. The LeakyReLU $(\cdot)$ function adds nonlinearity to the attention mechanism, which helps the model learn complex mappings from the input samples.

By utilizing this attention mechanism, GAT assigns different importance to nodes based on their relations, granting a refined understanding of the underlying steel plate data. GAT enables utilization of graph structure and node features simultaneously.

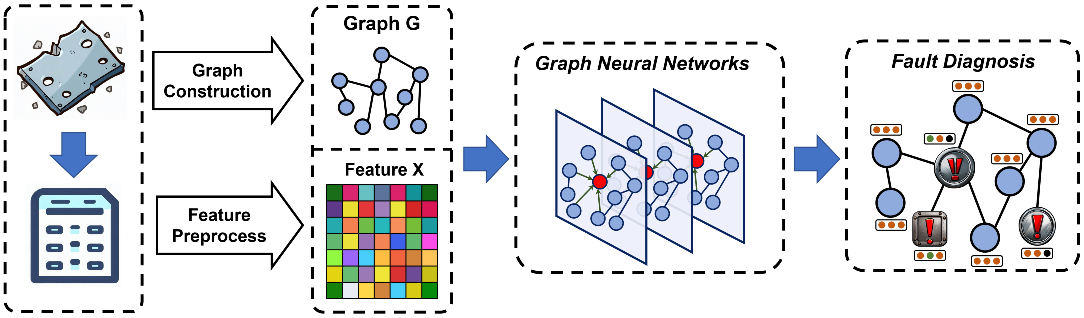


Fig. 2: Workflow of the Graph Construction with GNNs for Industrial Fault Diagnosis

Graph Isomorphism Networks: GIN is a salient extension in the GNN landscape, designed to address the graph isomorphism challenge. Instead of employing linear transformation like GCN, GIN integrates a multilayer perceptron for nodes, thereby imparting a non-linear transformation and enhancing model expressiveness. The pivotal equation governing GIN is:

$$h_i^{(l+1)} = \sigma\left(\left(1+\epsilon^{(l)}\right)h_i^{(l)} + \sum_{j\in\mathcal{N}(i)} W^{(l)}h_j^{(l)}\right) \tag{16}$$

Within Eq. (16), $\epsilon^{(l)}$ emerges as a layer-specific learnable parameter, adeptly balancing a node's features relative to its adjacent entities. Meanwhile, $W^{(l)}$ represents the layer-specific weight matrix.

The strengths of GIN in fault detection are rooted in its capacity to capture substructural patterns, leveraging information from both immediate and broader graph communities. This ability allows it to unearth fault features that may remain latent under other models.

The selected GNN models, GCN, GAT, and GIN, adopt distinctive approaches to graph data analysis, ensuring a comprehensive evaluation of A-SSGC across diverse graph processing contexts. GCN is leveraged for its strength in capturing local node features, GAT for its attention mechanisms that identify relevant graph segments, and GIN for its capacity to model complex topologies and higher-order interactions. This selection validates A-SSGC's theoretical soundness and practical applicability across various GNN architectures.

E. Computational Complexity Analysis

A computational analysis of the A-SSGC algorithm is performed to evaluate its efficiency. The analysis considers both time and space complexity. The variables in this analysis are the number of samples N, the feature dimensionality D, and the number of base graph construction methods n.

1) Time Complexity Analysis.: The overall time complexity is determined by the sum of three main stages: base graph construction, weight calculation, and adaptive sparsification. The first stage's complexity is defined by the most demanding base method. Methods such as k-NN, Cosine Similarity, and Euclidean Distance have a time complexity of $O(N^2D)$. The MST construction also requires an initial distance calculation of $O(N^2D)$, which dominates its process. In contrast, the Mahalanobis distance method is more computationally intensive, with a complexity derived from covariance matrix computation ($O(ND^2)$), matrix inversion ($O(D^3)$), and pairwise distance calculations ($O(N^2D^2)$).

The second stage is weight calculation. This step computes the pairwise similarity between the n base matrices using the Frobenius norm, which has a time complexity of $O(n^2N^2)$. The final stage is adaptive sparsification. This process operates on the fused matrix to prune edges, which requires $O(N^2)$ time. Since n is a small constant, the overall time complexity is dominated by the most expensive step in the base graph construction. This results in the following total time complexity:

$$T(N,D) = O(ND^2 + D^3 + N^2D^2) \tag{17}$$

2) Space Complexity Analysis.: The space complexity, denoted as $S(N,D)$, is determined by the memory required for the largest data structures. The algorithm stores the input feature matrix ($O(ND)$), the covariance matrix ($O(D^2)$), and n adjacency matrices ($O(N^2)$ each). Since n is a constant, the overall space complexity is dominated by the storage of the adjacency matrices, resulting in:

$$S(N,D) = O(N^2 + ND + D^2) \tag{18}$$

The complexity analysis indicates that A-SSGC has high computational requirements, particularly for large-scale datasets. The time complexity, driven by the Mahalanobis distance calculation, and the $O(N^2)$ space complexity can present practical challenges. Future research should focus on optimization. Potential directions include using approximation algorithms for the covariance matrix, parallelizing distance calculations, and exploring implicit graph representations that avoid storing the full adjacency matrix. Such advancements would enhance the algorithm's scalability and efficiency for large-scale industrial data analysis.

IV. Experiments & Discussion

A. Dataset Preprocessing & Model Evaluation

Dataset Descriptions. To evaluate A-SSGC's capability in capturing underlying material structures, we employ three public industrial datasets. The UCI and Kaggle Steel Plates Fault datasets were selected to analyze physical commonalities in materials with different scales and distributions. To assess performance on chemical commonalities, we utilized the UCI Glass Identification dataset, which describes samples based on their chemical composition. A detailed summary of the key statistics for these datasets, including sample counts, feature dimensions, and class distributions, is provided in Table I.

TABLE I: Summary of Dataset Statistics

Dataset	Samples	Features	Classes	Imbalance Ratio
UCI Steel Plates	1,941	27	7	4.49:1
Kaggle Steel Plates	19,218	27	7	1.35:1
UCI Glass	214	9	6	3.45:1

Dataset Processing. We performed several preprocessing steps to prepare the data. An initial analysis confirmed that the selected public datasets do not contain any missing values. Subsequently, all input features were standardized using Z-score normalization to transform the features to have a mean of zero and a standard deviation of one. Class imbalance in the two steel plate datasets was addressed using the Synthetic Minority Over-sampling Technique [62] to balance the training data, while the glass dataset retains its original distribution to isolate the impact of physicochemical modelling from oversampling artefacts. All datasets were split into 80% for training and 20% for testing.

Implementation Details. To ensure a fair and reproducible comparison across all graph construction methods, the downstream GNN classifiers were configured with a consistent architecture and fixed hyperparameters for each dataset. All GNNs utilized the same hidden dimension, learning rate and dropout rate, and were trained for 500 epochs. Specifically, the GCN consisted of two graph convolution layers with a ReLU activation function. The GAT was composed of two layers, utilizing 8 attention heads in the first layer. The GIN model was implemented with two layers, each containing a two-layer MLP for feature transformation.

For the base graph construction methods within A-SSGC, we used a consistent parameter setting. The number of k for k-NN was fixed at 10. For Cosine Similarity, Euclidean Distance, and Mahalanobis Distance, the connection threshold was uniformly set to the 10th percentile of the calculated similarity or distance distribution. This approach leaves only the two most critical hyperparameters of the A-SSGC fusion stage to be tuned: the base threshold θ_{base} and the degree factor δ. These two parameters were systematically optimized using the Optuna framework.

Evaluation Matrics. Performance is evaluated using four metrics: Accuracy $\left(\frac{TP+TN}{TP+TN+FP+FN}\right)$, Macro Precision $\left(\frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i+FP_i}\right)$, Macro Recall $\left(\frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i+FN_i}\right)$, and Macro F1-score $\left(\frac{2}{C}\sum_{i=1}^{C}\frac{\text{Precision}_i\times\text{Recall}_i}{\text{Precision}_i+\text{Recall}_i}\right)$, where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, C is the number of classes, and i indexes each class. Macro averaging computes metrics independently for each class and averages them, ensuring equitable evaluation across unbalanced industrial datasets where all types of defects are critical to diagnose accurately.

B. Comparison of Graph Construction Methods

In this section, we compare various graph construction methods, including basic approaches and the proposed A-SSGC. We also introduce three advanced baseline algorithms: GraphVAE [21], kNN-LDS [20], and Self-Adaptive [22]. These baselines were selected for their specific focus on constructing graph structures directly from feature data in unsupervised or semi-supervised paradigms. This approach is suitable for industrial scenarios where labeled data is often limited. Other notable graph learning models serve fundamentally different purposes. For example, Graph Sample and Aggregate(GraphSAGE) [63] is a GNN architecture that operates on a pre-existing graph. Methods like GRAPH Random Neural Networks(GRAND) [64] and Graph Diffusion Convolution(GDC) [65] are designed to refine or augment an existing graph structure, not to create one from the initial feature data. Furthermore, a Differentiable Graph Pooling(DiffPool) [66] is intended for graph-level tasks, which differ from our node-level objective. This focused selection of baselines provides a rigorous and relevant evaluation for industrial fault diagnosis.

To ensure a model-agnostic evaluation, all graph construction methods are benchmarked using three downstream GNN classifiers: GCN, GAT, and GIN. While GraphVAE and kNN-LDS originally rely on GCN, and Self-Adaptive approaches are tied to specific GNNs, we extend their implementations where necessary to support all three classifiers, mitigating biases introduced by classifier-specific strengths. The modified code is publicly available in our GitHub repository to ensure reproducibility. We tune each baseline's hyperparameters using Optuna [67], selecting configurations that yield optimal validation scores. The comparison, detailed in Table II, uses key metrics such as accuracy, precision, recall, and F1-score.

The experimental results show significant differences in the performance of graph construction methods for industrial fault diagnosis, which are determined by their ability to model inherent inter-sample relationships. Basic graph construction methods like k-NN and MST demonstrate competence in structured datasets with low feature heterogeneity. For instance, MST achieves strong performance on the UCI Steel Plates Fault dataset (F1 = 0.907 with GCN), where global connectivity aligns with its sparse edge selection strategy. k-NN also exhibits robustness, particularly with GIN classifiers (e.g., F1 = 0.784 on UCI Glass), leveraging localized feature alignment. However, these methods degrade significantly in complex industrial environments. Cosine Similarity and Euclidean Distance, reliant on direct feature comparisons, struggle to adapt to nonlinear interactions, as evidenced by Cosine Similarity's substantial F1-score deficit (0.630 vs. A-SSGC's 0.774 with GCN) on the Kaggle Steel Plates dataset. Mahalanobis Distance further struggles (Kaggle: F1 = 0.635 with GCN), its covariance-based metric appearing less suited to the non-Gaussian, multi-modal distributions often found in industrial data.

Deep graph construction methods also face significant challenges in these industrial applications. GraphVAE, primarily designed for generating graphs like molecular structures, performs poorly in these high-dimensional, noisy environments

TABLE II: Comparison of Graph Construction Methods

GNNs	Graph Constructions	UCI Steel Plate				Kaggle Steel Plate				UCI Glass			
		Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
GCN	k-NN	0.896	0.890	0.894	0.892	0.740	0.716	0.739	0.722	0.791	0.854	0.726	0.755
	Cosine	0.844	0.843	0.842	0.836	0.648	0.624	0.648	0.630	0.698	0.728	0.756	0.731
	Euclidean	0.871	0.864	0.868	0.864	0.645	0.629	0.648	0.627	0.767	0.773	0.751	0.750
	Mahalanobis	0.642	0.627	0.642	0.631	0.647	0.632	0.647	0.635	0.744	0.810	0.759	0.762
	MST	0.911	0.906	0.908	0.907	0.761	0.746	0.761	0.749	0.674	0.680	0.777	0.697
	GraphVAE	0.142	0.020	0.143	0.036	0.298	0.252	0.412	0.259	0.256	0.043	0.167	0.068
	kNN-LDS	0.704	0.784	0.704	0.727	0.612	0.663	0.612	0.625	0.744	0.633	0.614	0.615
	Self-Adaptive	0.791	0.787	0.792	0.789	0.552	0.489	0.481	0.460	0.465	0.486	0.675	0.520
	SSGC	0.915	0.913	0.915	0.914	0.762	0.742	0.762	0.748	0.814	0.806	0.802	0.812
	A-SSGC	**0.926**	**0.924**	**0.926**	**0.925**	**0.783**	**0.769**	**0.782**	**0.774**	**0.907**	**0.913**	**0.890**	**0.898**
GAT	k-NN	0.905	0.901	0.903	0.898	0.707	0.675	0.707	0.675	0.791	0.805	0.767	0.777
	Cosine	0.819	0.806	0.815	0.807	0.637	0.635	0.637	0.634	0.814	0.840	0.789	0.790
	Euclidean	0.894	0.888	0.891	0.887	0.607	0.625	0.608	0.603	0.651	0.700	0.725	0.691
	Mahalanobis	0.724	0.701	0.719	0.699	0.614	0.605	0.615	0.595	0.628	0.743	0.756	0.671
	MST	0.912	0.909	0.910	0.905	0.655	0.629	0.654	0.617	0.767	0.771	0.798	0.780
	GraphVAE	0.504	0.413	0.498	0.430	0.222	0.190	0.346	0.174	0.535	0.383	0.489	0.374
	kNN-LDS	0.706	0.831	0.705	0.737	0.633	0.681	0.633	0.647	0.698	0.840	0.657	0.705
	Self-Adaptive	0.783	0.787	0.782	0.763	0.538	0.566	0.473	0.446	0.302	0.520	0.561	0.391
	SSGC	0.943	0.940	0.941	0.940	0.745	0.726	0.744	0.713	0.837	0.830	0.831	0.823
	A-SSGC	**0.944**	**0.944**	**0.944**	**0.943**	**0.786**	**0.769**	**0.785**	**0.775**	**0.884**	**0.905**	**0.937**	**0.915**
GIN	k-NN	0.908	0.902	0.906	0.903	0.780	0.766	0.780	0.770	0.814	0.871	0.771	0.784
	Cosine	0.819	0.812	0.815	0.811	0.625	0.592	0.626	0.600	0.837	0.824	0.796	0.803
	Euclidean	0.912	0.908	0.910	0.908	0.578	0.559	0.579	0.555	0.791	0.784	0.740	0.752
	Mahalanobis	0.679	0.653	0.679	0.655	0.677	0.653	0.676	0.654	0.721	0.739	0.707	0.704
	MST	0.919	0.915	0.917	0.915	0.818	0.801	0.817	0.806	0.744	0.714	0.756	0.727
	GraphVAE	0.404	0.417	0.558	0.376	0.202	0.138	0.141	0.131	0.558	0.572	0.705	0.575
	kNN-LDS	0.877	0.879	0.877	0.877	0.707	0.704	0.706	0.704	0.814	0.837	0.763	0.787
	Self-Adaptive	0.880	0.877	0.880	0.878	0.542	0.512	0.532	0.501	0.605	0.573	0.628	0.578
	SSGC	0.918	0.917	0.918	0.916	0.823	0.815	0.822	0.817	0.814	0.848	0.825	0.826
	A-SSGC	**0.930**	**0.929**	**0.930**	**0.929**	**0.847**	**0.836**	**0.846**	**0.840**	**0.907**	**0.889**	**0.949**	**0.909**

Note: Acc = Accuracy, Prec = Precision, Rec = Recall. Bold indicates best results.

(e.g., Kaggle: F1 = 0.259 with GCN). Its reliance on global latent variables appears insufficient for constructing the sparse and meaningful inter-sample connections critical for industrial commonality discovery. kNN-LDS, despite integrating semi-supervised edge refinement, shows limited and inconsistent improvements over traditional baselines. In the Kaggle dataset with GCN, kNN-LDS achieves an F1-score of 0.625, a 14.9% deficit compared to A-SSGC's 0.774. While kNN-LDS marginally exceeds k-NN in one specific scenario (UCI Glass with GIN: F1 = 0.787 vs. 0.784), this small gain is not consistent across datasets and classifiers. The Self-Adaptive method, which aims to learn graph structure from node embeddings, also demonstrates limitations. For instance, with GCN, it yields an F1-score of only 0.460 on the Kaggle dataset and 0.520 on the UCI Glass. This underperformance suggests that learning highly discriminative embeddings for graph construction is challenging. The task becomes particularly difficult when using complex, noisy industrial features without guidance from multiple similarity perspectives. Such methods may struggle if the learned embedding space does not adequately capture the multifaceted physicochemical nuances critical for fault diagnosis.

The proposed A-SSGC consistently overcomes the limitations of existing methods by adaptively integrating complementary graph construction strategies. A-SSGC fuses multiple fundamental perspectives, including local proximity (e.g., k-NN), global connectivity (e.g., MST), and additional statistical or structural metrics. This comprehensive fusion allows A-SSGC to capture diverse and complementary views of inter-sample relationships, leading to advanced performance across tasks. For example, on the Kaggle dataset with GCN, A-SSGC improves the F1-score by 31.5% over Self-Adaptive (0.774 vs. 0.460) and by 51.5% over GraphVAE (0.774 vs. 0.259). Its adaptive sparsification strategy further prunes redundant edges while preserving meaningful structural links. On the UCI Glass dataset, A-SSGC reaches an F1-score of 0.898 with GCN, surpassing its predecessor SSGC by 8.6%. This gain demonstrates the value of replacing static thresholding with adaptive fusion. Furthermore, A-SSGC maintains robustness with diverse GNN classifiers. For example, on the UCI Steel Plate dataset, it achieves consistently high F1-scores of 0.925 with GCN, 0.943 with GAT, and 0.929 with GIN. This strong and stable performance across different GNN backbones enables it to significantly surpass competing methods like k-NN and Self-Adaptive, particularly on challenging datasets such as Kaggle.

These results highlight two key observations. First, methods that rely on asingle fixed heuristics (e.g., k-NN, MST), direct feature comparisons (e.g., Cosine, Euclidean) fail to capture the nonlinearity, noise, and structural diversity of industrial data. Even deep models like GraphVAE, Self-Adaptive, and kNN-LDS struggle when they do not explicitly incorporate diverse similarity views. Second, A-SSGC's strength lies in its multi-strategy fusion and adaptive sparsification. While kNN-

LDS relies on semi-supervised refinement and requires labeled data, A-SSGC is fully unsupervised and avoids the limitations imposed by heuristic initialization. Self-Adaptive approaches, although flexible, may underperform if their learned embeddings do not adequately reflect all relevant affinities. In contrast, A-SSGC constructs the graph by jointly considering multiple similarity paradigms and refining the structure in a data-driven, adaptive manner. This design enables robust performance gains in both structured and noisy settings. Its consistent superiority across datasets and GNN architectures underscores its value as a general-purpose, robust solution for industrial fault diagnosis in heterogeneous environments.

C. Runtime and Efficiency Analysis

Computational efficiency is a critical factor for the practical deployment of industrial systems. This section evaluates the runtime of the compared graph construction methods to assess their feasibility. Table III presents the results, grouping the methods into three distinct categories for a clear analysis: basic methods, deep graph construction methods, and the proposed methods.

The results show a clear difference in computational cost between the categories. The basic methods are very fast on small-scale data. For instance, Euclidean Distance completes in only 0.004 seconds on the UCI Glass dataset. However, the scalability of some basic methods is a significant weakness. The runtime for both Mahalanobis Distance and MST increases dramatically on the larger Kaggle dataset. Mahalanobis Distance increases from 0.724 seconds to 70.8 seconds, while MST increases from 0.647 seconds to 101.4 seconds. In contrast, the deep graph construction methods require the most time. GraphVAE and Self-Adaptive need over 2,800 seconds on the Kaggle dataset, making them impractical for many applications. Even the more efficient kNN-LDS requires a substantial 236.6 seconds on the same dataset.

The proposed A-SSGC provides a balance between efficiency and performance. A-SSGC is computationally more intensive than the basic methods. Its runtime of 2.703 seconds on the UCI Steel Plate dataset is notably higher than the 0.285 seconds required by k-NN. However, its main advantage is a significant speed improvement over the deep graph construction approaches. On the Kaggle dataset, A-SSGC is more than 13 times faster than Self-Adaptive, completing in 215.4 seconds. This suggests a critical trade-off. The increased runtime of A-SSGC over basic methods is an affordable cost when considering its superior diagnostic performance. By avoiding the extreme computational time of other deep methods, A-SSGC proves to be a practical and robust solution for industrial fault diagnosis.

D. Comparison of Classification Models

This study comprehensively evaluates various classification methods, including traditional ML models and GNNs. Through a detailed comparison of experimental results, we provide an in-depth analysis of each model's performance in industrial fault diagnosis tasks. Model performance is assessed based on accuracy, precision, recall, and F1 scores, as shown in Table IV.

In the UCI Steel Plate Faults dataset, classification performance improves progressively with the enhancement of the models' nonlinear fitting capabilities. LR, being a linear model, exhibits relatively limited performance, achieving an accuracy of 82.7% and an F1-score of 81.9%. The Decision Tree model captures nonlinear features, reaching an accuracy of 85.5% and an F1-score of 85.4%. The k-NN algorithm, which relies on distance metrics between samples, attains an accuracy of 89.1% and an F1-score of 88.3%, performing well on datasets with clear physical characteristics. The MLP, with its strong nonlinear fitting ability, achieves an accuracy of 90.0% and an F1-score of 89.8%.

However, in the Kaggle Steel Plate Faults dataset, which contains noise and synthetic samples, SVM and MLP exhibit high sensitivity to noise, with MLP's F1-score dropping to 60.4%. In contrast, k-NN and Decision Tree demonstrate stronger robustness, maintaining F1-scores of 70.6% and 71.2%, respectively. Notably, while k-NN performs relatively well, it delivers the worst performance in the UCI Glass Identification dataset, with an F1-score of only 42.4%. This indicates that models based on simple distance metrics cannot capture chemical commonalities. Additionally, Naive Bayes, which relies on the assumption of feature independence, performs poorly across all datasets. These results suggest the presence of strong correlations rooted in material properties, and the dependence of traditional ML models on the independent and i.i.d. assumption limits their ability to capture these commonalities.

Compared to traditional ML models, the A-SSGC combined with GNNs shows significant performance improvements, partially mitigating the limitations of the i.i.d. assumption. In the UCI Steel Plate Faults dataset, the GAT(A-SSGC) leverages attention mechanisms to capture critical connections within the graph, achieving an accuracy of 94.4% and an F1-score of 94.3%, significantly outperforming all traditional models. In the Kaggle Steel Plate Faults dataset with synthetic samples and noise, the GIN(A-SSGC) attains an F1-score of 84.0%, improving by 18% over the best-performing traditional model—Decision Tree. A-SSGC also demonstrates excellent adaptability when handling imbalanced data. In the unsampled UCI Glass Identification dataset, A-SSGC achieves an accuracy of 90.7% and an F1-score of 90.9%, outperforming traditional ML models by up to 31.3 %. These results indicate that A-SSGC effectively captures intrinsic connections between samples by constructing precise graph structures, enhancing classification performance and model robustness.

The integration of A-SSGC with GNNs not only enhances fault diagnosis accuracy but also expands the application potential of GNNs to data with implicit relationships. By constructing high-quality graph structures, A-SSGC enables models to better leverage the intrinsic structures within the data. Transforming tabular data into graph representations, A-SSGC not only broadens the applicability of GNNs but also offers an innovative approach for uncovering inherent patterns in tabular data. This method exhibits significant advantages in handling complex industrial datasets, providing new insights

TABLE III: Runtime Comparison of Graph Construction Methods (seconds)

Graph Construction Method	UCI Steel Plate	Kaggle Steel Plate	UCI Glass
k-NN	0.285	0.436	0.066
Cosine Similarity	0.171	17.234	0.024
Euclidean Distance	0.189	20.288	0.004
Mahalanobis Distance	0.724	70.803	0.197
MST	0.647	101.404	0.006
GraphVAE	45.656	2807.866	1.787
kNN-LDS	7.113	236.629	3.124
Self-Adaptive	41.449	2876.427	7.415
SSGC	2.274	205.503	0.478
A-SSGC	2.703	215.382	0.712

TABLE IV: Comparison of Different Classification Models

Models	UCI Steel Plate				Kaggle Steel Plate				UCI Glass			
	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
LR	0.827	0.818	0.827	0.819	0.697	0.674	0.697	0.675	0.721	0.686	0.617	0.629
k-NN	0.891	0.891	0.890	0.883	0.744	0.717	0.743	0.706	0.628	0.419	0.452	0.424
MLP	0.900	0.897	0.900	0.898	0.623	0.603	0.624	0.604	0.744	0.696	0.591	0.595
SVM	0.890	0.888	0.890	0.885	0.692	0.669	0.692	0.674	0.721	0.663	0.570	0.581
Naive Bayes	0.734	0.747	0.735	0.731	0.567	0.570	0.567	0.567	0.558	0.605	0.628	0.593
Decision Tree	0.855	0.853	0.855	0.854	0.715	0.710	0.714	0.712	0.721	0.723	0.721	0.697
GCN(A-SSGC)	0.926	0.924	0.926	0.925	0.783	0.769	0.782	0.774	**0.907**	**0.913**	0.890	0.898
GAT(A-SSGC)	**0.944**	**0.944**	**0.944**	**0.943**	0.786	0.769	0.785	0.775	0.884	0.905	0.937	**0.915**
GIN(A-SSGC)	0.930	0.929	0.930	0.929	**0.847**	**0.836**	**0.846**	**0.840**	**0.907**	0.889	**0.949**	0.909

Note: Acc = Accuracy, Prec = Precision, Rec = Recall. Bold indicates best results.

for future research in industrial fault diagnosis and related fields.

E. Ablation and Sensitivity Analyses

In this subsection, we perform comprehensive ablation and sensitivity analyses to clearly understand the contributions of key components and critical hyperparameters in the A-SSGC model.

1) Ablation Study on Key Components: To thoroughly examine the individual contributions of critical components within A-SSGC, we evaluate four variants: (1) retaining only self-connections without any graph construction learning ("Only Self-connected Graph"); (2) omitting the sparsification strategy completely, equivalent to SSGC ("w/o Sparsification Strategy"); (3) removing the adaptive mechanism based on node degrees ("w/o degree factor δ"); and (4) removing the base threshold θ_{base} ("w/o base threshold θ_{base}").Detailed performance comparisons are presented in Table V.

Table V demonstrates the importance of each component. The "Only Self-connected Graph" variant, representing a baseline without graph learning or sparsification, performed moderately, indicating that relying solely on node features is insufficient for effective fault identification. Removing the adaptive sparsification strategy improved performance over the baseline but remained significantly lower than the complete model. This highlights the general benefit of the adaptive sparsification mechanism. Analyzing the specific components of this strategy further reveals their importance. Removing the degree factor δ caused a substantial performance drop across all datasets, most notably on the Kaggle Steel Plate dataset, where the F1-score fell sharply from 0.774 to 0.498. This shows that adapting the threshold based on node degrees is crucial to distinguish edges effectively, particularly in complex datasets. Similarly, removing the θ_{base} also led to significant performance degradation, F1-scores dropped to 0.502 on Kaggle and 0.706 on UCI Glass, confirming the need for a baseline filtering level to maintain graph integrity.

Overall, the complete A-SSGC model consistently achieved the best performance. These results confirm that both the synergistic fusion approach and the adaptive sparsification strategy, driven by the interplay of the degree factor δ and the base threshold θ_{base}, are indispensable. They work together to effectively capture the shared characteristics among industrial material samples and enhance the performance of fault diagnosis.

2) Hyperparameter Sensitivity Study: We further evaluated the sensitivity of A-SSGC's performance to its two key hyperparameters: the degree factor δ and the base threshold θ_{base}. The experiments were conducted on the UCI Glass dataset using GCN as the downstream classifier, with results shown in Figure 3.

Figure 3a illustrates the model's sensitivity to the degree factor δ, tested within the range [0,2]. The model's performance peaks around $\delta = 0.35$ (Accuracy=0.860, F1-score=0.839). Very small δ values, approaching a fixed threshold scenario ($\delta = 0$, F1 = 0.741), result in suboptimal performance, indicating the benefit of considering node degrees. Conversely, very large δ values ($\delta = 2$, F1$\approx$ 0.759) also lead to decreased performance, potentially making the threshold overly sensitive to degree variations. The results suggest that a moderate δ value between 0.2 and 0.35 optimally balances global thresholding with degree-based adaptation.

TABLE V: Ablation Study on Core Components of A-SSGC

Variants	UCI Steel Plate				Kaggle Steel Plate				UCI Glass			
	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
Only Self-connected Graph	0.879	0.876	0.879	0.877	0.682	0.661	0.682	0.666	0.744	0.736	0.741	0.722
w/o Sparsification Strategy	0.915	0.913	0.915	0.914	0.762	0.742	0.762	0.748	0.814	0.806	0.802	0.812
w/o degree factor δ	0.856	0.852	0.856	0.852	0.550	0.531	0.504	0.498	0.744	0.759	0.739	0.741
w/o base threshold θ_{base}	0.863	0.860	0.863	0.860	0.553	0.534	0.508	0.502	0.721	0.723	0.727	0.706
Complete Model	**0.926**	**0.924**	**0.926**	**0.925**	**0.783**	**0.769**	**0.782**	**0.774**	**0.907**	**0.913**	**0.890**	**0.898**

Note: Acc = Accuracy, Prec = Precision, Rec = Recall. Bold indicates best results.

Figure 3b shows performance sensitivity to the base threshold θ_{base}, tested within the range $[0, 1]$. The F1-score achieves its primary peak at $\theta_{base} = 0.35$ (F1 = 0.864) and a strong secondary peak near $\theta_{base} = 0.2$ (F1 = 0.844). Performance drops sharply as $\theta_{base} = 0.35$ approaches 0 (F1 = 0.706), highlighting the need for a baseline filter to remove noisy or weak edges. Performance also gradually declines for θ_{base} values above the optimal range (F1 ≈ 0.754 at $\theta_{base} = 1$), likely due to filtering out useful connections. The model exhibits relatively stable and near-optimal performance for θ_{base} values between 0.2 and 0.4, indicating reasonable robustness within this range.

Overall, the sensitivity analysis confirms that δ and θ_{base} significantly influence A-SSGC's performance, but the model remains robust within specific ranges around the optimal values. This reinforces the effectiveness of the adaptive sparsification strategy and underscores the importance of appropriately tuning these parameters to construct high-quality graphs capable of capturing relevant physicochemical commonalities.

F. Constructed Graph Visualization

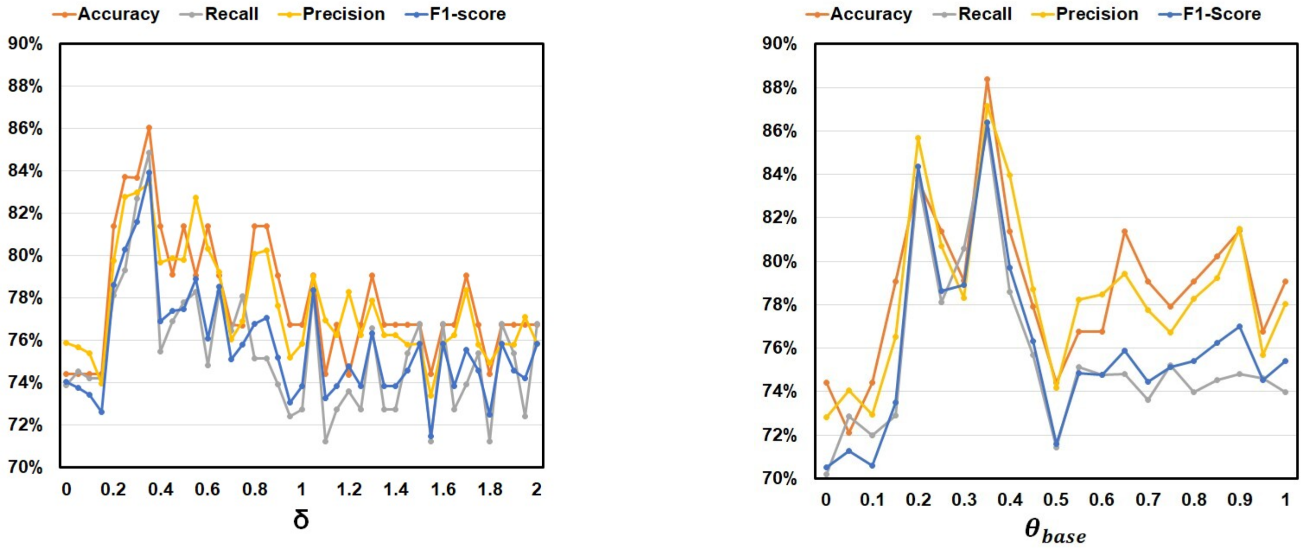
According to [68], steel plate samples with identical physical defects share measurable characteristics such as shape, size, and surface texture anomalies derived from manufacturing processes. These properties are encoded as numerical features, and their similarity guides graph construction to model physical commonalities spatially. The UCI Steel Plates Fault dataset categorizes defects into distinct types: Pastry, Z_Scratch, K_Scratch, and Bumps, each with unique physical origins. Pastry defects manifest as rounded material displacements, Z_Scratch as linear abrasions from sharp objects, K_Scratch as irregular chaotic scratches, and Bumps as localized surface deformations.

Table VI visualizes the graphs constructed using various methods. Basic graph construction methods exhibit distinct limitations. Cosine Similarity and Euclidean Distance produce fragmented graphs for Z_Scratch and K_Scratch defects, with numerous isolated nodes indicating incomplete relationship modelling. These methods struggle to adapt to nonlinear feature interactions, particularly in defects with irregular morphologies. While k-NN improves local connectivity for Pastry and Bumps through proximity-based clustering, it fails to establish global relationships critical for Z/K_Scratch defects. MST enforces global continuity but homogenizes graph structures across defect types; for instance, Pastry and Bumps defects yield nearly identical spanning trees despite their divergent physical origins. Such rigidity undermines the representation of unique physicochemical signatures.

Deep graph construction methods struggle to adapt to industrial data's complexity. GraphVAE generates overly dense topologies approaching complete connectivity, where nodes cluster indiscriminately within defect categories. This lack of sparsity obscures critical relationships, as trivial edges drown out discriminative physicochemical patterns. kNN-LDS, another deep baseline, initialises with a k-NN graph and refines edges via semi-supervised updates. However, visual analysis reveals limited structural improvements over its k-NN foundation. In Bump defects, kNN-LDS retains k-NN's local clusters but fails to establish meaningful connections between subclusters sharing physicochemical similarities. For Z_Scratch defects, edge updates introduce sparse long-range links that lack geometric coherence, suggesting inadequate adaptation to nonlinear feature interactions. Similarly, the Self-Adaptive method, which learns graph structures directly from node embeddings, yields highly fragmented and structurally incoherent topologies, particularly for Z_Scratch and K_Scratch defects. The resulting disjointed clusters suggest that its learned embedding space fails to capture the coherent physical properties defining these fault types, underscoring its instability when tasked with constructing graphs from complex industrial features without explicit multi-metric guidance. These visual limitations correlate with its underwhelming performance(e.g., F1 = 0.727 vs. k-NN's 0.892 with GCN), highlighting the challenge of rectifying initialization biases through incremental refinements.

SSGC addresses these issues by combining multiple construction principles, eliminating isolated nodes and producing defect-specific structures. However, it introduces redundant connections visible as dense inter-cluster edges in Pastry graphs, which obscure salient relationships. A-SSGC resolves this via adaptive sparsification, dynamically pruning non-discriminative edges while preserving physicochemical coherence. For example, in K_Scratch defects, A-SSGC selectively connects samples with overlapping scratch orientation and depth features, whereas SSGC links nodes indiscriminately. This refinement yields sparser yet semantically richer graphs, aligning with A-SSGC's superior diagnostic accuracy (e.g., F1 = 0.925 vs. SSGC's 0.914 in UCI Steel Plates).

These visual patterns collectively demonstrate that industrial-grade graph construction requires multi-strategy fusion guided by domain-specific adaptability. While SSGC overcomes fragmentation and homogeneity issues of



(a) Sensitivity to degree factor δ

(b) Sensitivity to base threshold θ_{base}

Fig. 3: Hyperparameter Sensitivity Analysis of A-SSGC (GCN Classifier)

basic/deep graph construction methods, A-SSGC's threshold optimization proves critical for balancing connectivity with interpretability—a topological refinement essential for precise fault diagnosis.

V. Conclusion

This paper introduces A-SSGC, a novel graph construction framework designed to overcome the challenges of inexplicit relational data in industrial fault diagnosis. A-SSGC outperforms traditional ML models, basic graph construction methods, and unsupervised/semi-supervised deep graph construction approaches, consistently demonstrating superior performance across multiple industrial datasets. The key advantage of A-SSGC lies in its adaptive fusion strategy. This strategy constructs graph structures that better capture local and global relationships. It is particularly effective in noisy and imbalanced data. These results highlight A-SSGC's potential for broader applications in industrial settings and beyond, particularly in cases where explicit relational data is unavailable. By enhancing interpretability and enabling reliable fault diagnosis, A-SSGC contributes to the advancement of graph-based learning techniques, offering significant promise for automation in industrial processes. Future work will address the primary limitations of A-SSGC: its computational overhead and sensitivity to key hyperparameters. We plan to explore approximation techniques for scalability and automated ML approaches for robust parameter optimization.

References

[1] N. R. Baddoo, "Stainless steel in construction: A review of research, applications, challenges and opportunities," Journal of Constructional Steel Research, vol. 64, no. 11, pp. 1199–1206, November 2008.

[2] C.-Y. Chen, F.-Y. Hung, T.-S. Lui, and L.-H. Chen, "Microstructures and mechanical properties of austempering sus440 steel thin plates," Metals, vol. 6, no. 2, p. 35, 2016.

[3] E. Axinte, "Glasses as engineering materials: A review," Materials & Design, vol. 32, no. 4, pp. 1717–1732, April 2011.

[4] A. Abid, M. T. Khan, and J. Iqbal, "A review on fault detection and diagnosis techniques: basics and beyond," Artif Intell Rev, vol. 54, no. 5, pp. 3639–3664, June 2021.

[5] D. Miljković, "Fault detection methods: A literature survey," 2011 Proceedings of the, vol. 34th International Convention MIPRO, pp. 750–755, May 2011.

[6] W. D. Callister et al., Materials science and engineering: an introduction, vol. 7. John wiley & sons. New York: John Wiley & Sons, 2007, vol. 7.

[7] S. Guo, J. Yu, X. Liu, C. Wang, and Q. Jiang, "A predicting model for properties of steel using the industrial big data based on machine learning," Computational Materials Science, vol. 160, pp. 95–104, April 2019.

[8] Y. Tian, M. Fu, and F. Wu, "Steel plates fault diagnosis on the basis of support vector machines," Neurocomputing, vol. 151, pp. 296–303, March 2015.

[9] T. Nkonyana, Y. Sun, B. Twala, and E. Dogo, "Performance evaluation of data mining techniques in steel manufacturing industry," Procedia Manufacturing, vol. 35, pp. 623–628, January 2019.

[10] J. Chen, "The application of tree-based ml algorithm in steel plates faults identification," Journal of Applied and Physical Sciences, vol. 4, no. 2, pp. 47–54, 2018.

[11] Ravinder et al., "Artificial intelligence and machine learning in glass science and technology: 21 challenges for the 21st century," International Journal of Applied Glass Science, vol. 12, no. 3, pp. 277–292, 2021.

[12] A. Kensert, R. Bouwmeester, K. Efthymiadis, P. V. Broeck, G. Desmet, and D. Cabooter, "Graph convolutional networks for improved prediction and interpretability of chromatographic retention data," Anal. Chem., vol. 93, no. 47, pp. 15 633–15 641, November 2021.

[13] Q. Liu, D. Luo, T. Wen, H. GholamHosseini, X. Qiu, and J. Li, "Poi-3dgcn: Predicting odor intensity of monomer flavors based on three-dimensionally embedded graph convolutional network," Expert Systems with Applications, vol. 199, p. 116997, August 2022.

[14] H. Cai, H. Zhang, D. Zhao, J. Wu, and L. Wang, "Fp-gnn: a versatile deep learning architecture for enhanced molecular property prediction," Briefings in bioinformatics, vol. 23, no. 6, p. bbac408, 2022.

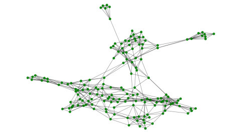
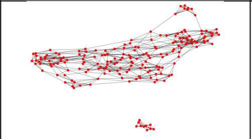
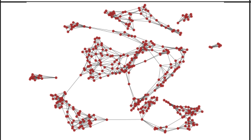
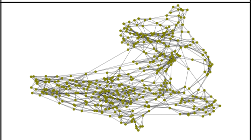
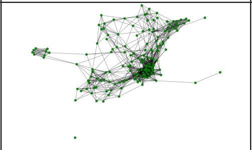
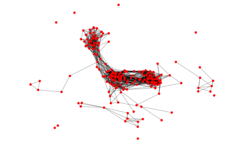
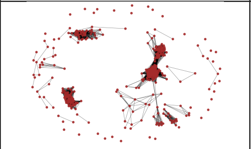
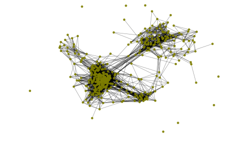
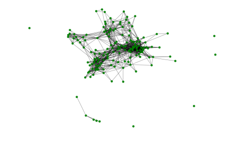
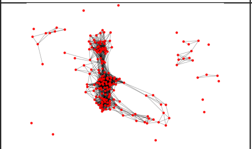
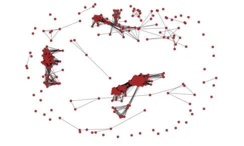
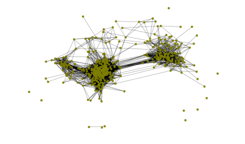
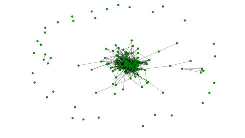
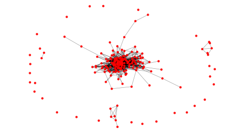
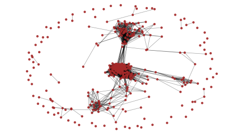
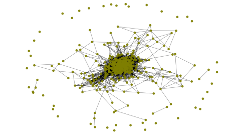
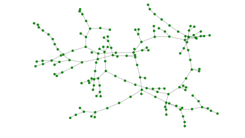
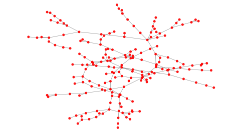
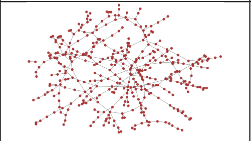
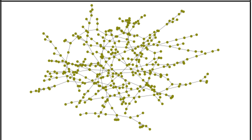
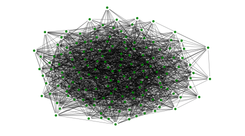
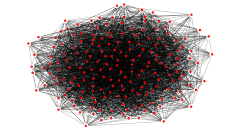
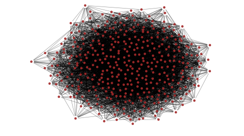
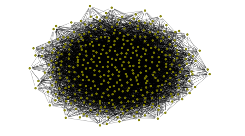
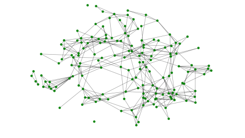
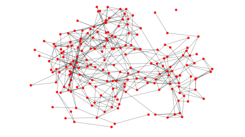
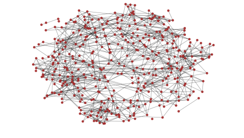
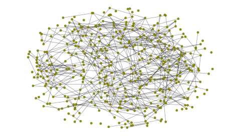
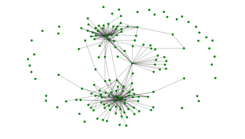
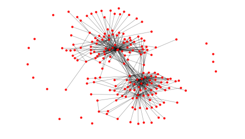
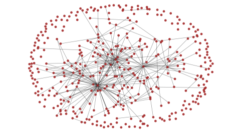
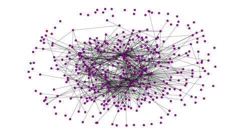
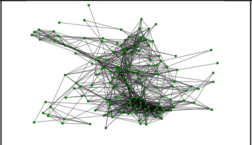
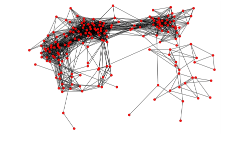
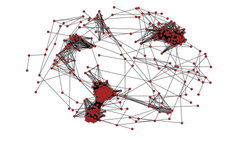
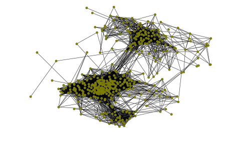
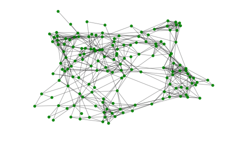
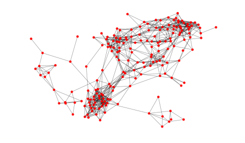
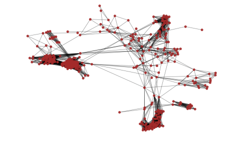
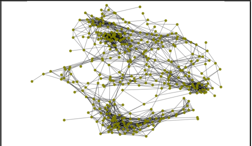
[15] J. Dong, C. Chen, C. Zhang, J. Ma, and K. Peng, "Knowledge graph embedding with graph convolutional network and bidirectional gated recurrent unit for fault diagnosis of industrial processes," IEEE Sensors Journal, vol. 25, no. 5, pp. 8611–8620, 2025.

[16] H. Zhu, S. Rho, S. Liu, and F. Jiang, "Learning spatial graph structure for multivariate kpi anomaly detection in large-scale cyber-physical systems," IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1–16, 2023.

[17] W.-M. Roth, "Emergence of graphing practices in scientific research," Journal of Cognition and Culture, vol. 4, no. 3-4, pp. 595–627, January 2004.

[18] S. Feiler, K. G. Müller, A. Müller, R. Dahlhaus, and W. Eich, "Us-

TABLE VI: Constructed Graphs of Steel Samples with Physical Defects

Graph Construction	Pastry	Z_Scratch	K_Scratch	Bumps
k-NN				
Cosine				
Euclidean				
Mahalanobis				
MST				
GraphVAE				
kNN-LDS				
Self-Adaptive				
SSGC				
A-SSGC				

ing interaction graphs for analysing the therapy process," Psychother Psychosom, vol. 74, no. 2, pp. 93–99, 2005.

[19] Y. Chen, Z. Chen, and H. U. Amin, "Lg-vgae: a local and global collaborative variational graph autoencoder for detecting crypto money laundering: Y. chen et al." Knowledge and Information Systems, pp. 1–24, 2025.

[20] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Learning discrete structures for graph neural networks," in International conference on machine learning. PMLR, 2019, pp. 1972–1982.

[21] M. Simonovsky and N. Komodakis, "Graphvae: Towards generation of small graphs using variational autoencoders," in Artificial Neural Networks and Machine Learning – ICANN 2018, V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, and I. Maglogiannis, Eds. Cham: Springer International Publishing, 2018, pp. 412–422.

[22] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, 2020, pp. 753–763.

[23] Y. Zhu, Y. Du, Y. Wang, Y. Xu, J. Zhang, Q. Liu, and S. Wu, "A survey on deep graph generation: Methods and applications," in Learning on

Graphs Conference. PMLR, 2022, pp. 47–1.

[24] R. Ghiasi, A. Bosaghzadeh, and H. Amirkhani, "Enhancing graph structure learning through multiple features and graphs fusion," Computers and Electrical Engineering, vol. 123, p. 110200, 2025.

[25] Y. Li, N. Zhong, D. Taniar, and H. Zhang, "Mcgnet+: an improved motor imagery classification based on cosine similarity," Brain Informatics, vol. 9, no. 1, p. 3, February 2022.

[26] Y. Zeng, X. Zhou, J. Rao, Y. Lu, and Y. Yang, "Accurately clustering single-cell rna-seq data by capturing structural relations between cells through graph convolutional network," in 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE, 2020, pp. 519–522.

[27] Y. Liu, B. Lang, and F. Quan, "Mst-hgcn: a minimum spanning tree hyperbolic graph convolutional network," Applied Intelligence, vol. 53, no. 11, pp. 14 515–14 526, 2023.

[28] Y. Chen, H. Zhu, and Z. Chen, "Multi-dgi: Multi-head pooling deep graph infomax for human activity recognition," Mobile Networks and Applications, pp. 1–12, 2024.

[29] Y. Chen, Z. Chen, and H. U. Amin, "Synergistic similarity graph construction (ssgc) for steel plate fault diagnosis with graph attention networks," in 2023 IEEE 6th International Conference on Knowledge Innovation and Invention (ICKII). IEEE, 2023, pp. 655–660.

[30] N. S. Gameiro and A. J. M. Cardoso, "A new method for power converter fault diagnosis in srm drives," IEEE Transactions on Industry Applications, vol. 48, no. 2, pp. 653–662, March 2012.

[31] X. Zhang, H. Wang, M. Ren, M. He, and L. Jin, "Rolling bearing fault diagnosis based on multiscale permutation entropy and soa-svm," Machines, vol. 10, no. 6, p. 485, 2022.

[32] M. V. Moreira and J.-J. Lesage, "Fault diagnosis based on identified discrete-event models," Control Engineering Practice, vol. 91, p. 104101, October 2019.

[33] D.-H. Ham and W. C. Yoon, "The training effects of principle knowledge on fault diagnosis performance," Human Factors and Ergonomics in Manufacturing & Service Industries, vol. 17, no. 3, pp. 263–282, 2007.

[34] H.-x. Hu, A.-l. Gehin, and M. Bayart, "An extended qualitative multi-faults diagnosis from first principles i: Theory and modelling," in Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference. IEEE, 2009, pp. 1008–1013.

[35] Y. Gao, M. Chen, and D. Yu, "Semi-supervised graph convolutional network and its application in intelligent fault diagnosis of rotating machinery," Measurement, vol. 186, p. 110084, December 2021.

[36] A. Sztyber, A. Ostasz, and J. M. Kościelny, "Graph of a process—a new tool for finding model structures in a model-based diagnosis," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 45, no. 7, pp. 1004–1017, July 2015.

[37] C. Zhang, J. Cui, and W. Liu, "Multilayer feature extraction of agcn on surface defect detection of steel plates," Computational Intelligence and Neuroscience, vol. 2022, p. e2549683, October 2022.

[38] W. Yu, W. Cheng, C. C. Aggarwal, H. Chen, and W. Wang, "Link prediction with spatial and temporal consistency in dynamic networks." in IJCAI, 2017, pp. 3343–3349.

[39] T. Roine et al., "Reproducibility and intercorrelation of graph theoretical measures in structural brain connectivity networks," Med Image Anal, vol. 52, pp. 56–67, February 2019.

[40] N. Wu et al., "A nonparametric approach to uncovering connected anomalies by tree shaped priors," IEEE Transactions on Knowledge and Data Engineering, vol. 31, no. 10, pp. 1849–1862, October 2019.

[41] G. Qin, S. Xu, D. Yao, and Z. Zhang, "Study on the degradation of mechanical properties of corroded steel plates based on surface topography," Journal of Constructional Steel Research, vol. 125, pp. 205–217, October 2016.

[42] H. Su, X. Luo, F. Chai, J. Shen, X. Sun, and F. Lu, "Manufacturing technology and application trends of titanium clad steel plates," J. Iron Steel Res. Int., vol. 22, no. 11, pp. 977–982, November 2015.

[43] M. L. Skryabin, "The study of influence of chemical composition of steel 35hgsl on the characteristics of shrinkage, casting defects and microstructure," J. Phys.: Conf. Ser., vol. 1515, no. 4, p. 042107, April 2020.

[44] S. Serajzadeh, "Modelling of temperature history and phase transformations during cooling of steel," Journal of Materials Processing Technology, vol. 146, no. 3, pp. 311–317, March 2004.

[45] P. Wu, Y. Wang, J. Gao, X. Zhang, S. Lou, and C. Yang, "Multichannel dynamic graph convolutional network-based fault diagnosis and its application in blast furnace ironmaking process," IEEE Sensors Journal, vol. 23, no. 23, pp. 29 293–29 302, 2023.

[46] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in Proceedings of the 5th International Conference on Learning Representations (ICLR 2017), Toulon, France, April 24–26 2017, conference Track Proceedings.

[47] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. (2018) Graph attention networks. In International conference on learning representations.

[48] K. Xu*, W. Hu*, J. Leskovec, and S. Jegelka. (2022) How powerful are graph neural networks? Presented at the International Conference on Learning Representations, Sep. 2018. Accessed: May 20.

[49] H. Kwon, Z. Du, and Y. Li, "Alphafold 2-based stacking model for protein solubility prediction and its transferability on seed storage proteins," International Journal of Biological Macromolecules, vol. 278, p. 134601, 2024.

[50] S. Wu, W. Zhang, F. Sun, and B. Cui, "Graph neural networks in recommender systems: A survey," ACM Computing Surveys, vol. 55, pp. 1–37, 2020.

[51] O. Wieder, M. Kuenemann, M. Wieder, T. Seidel, C. Meyer, S. D. Bryant, and T. Langer, "Improved lipophilicity and aqueous solubility prediction with composite graph neural networks," Molecules, vol. 26, no. 20, p. 6185, 2021.

[52] J. Wu, Z. Chen, S. Xiao, Z. Liu, and Z. Hu, "Deepmoic: multi-omics data integration via deep graph convolutional networks for cancer subtype classification," BMC Genomics, vol. 25, no. 1, p. 1209, 2024.

[53] B. Fatemi, L. El Asri, and S. M. Kazemi, "Slaps: Self-supervision improves structure learning for graph neural networks," Advances in Neural Information Processing Systems, vol. 34, pp. 22 667–22 681, 2021.

[54] W. Huang, G. Wan, M. Ye, and B. Du, "Federated graph semantic and structural learning," in Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23, 2023, pp. 3830–3838.

[55] O. Anava and K. Levy. (2024) k -nearest neighbors: From global to local. In Advances in Neural Information Processing Systems, Curran Associates, Inc., 2016. Accessed: Apr. 11.

[56] B. Li and L. Han, "Distance weighted cosine similarity measure for text classification," in IDEAL 2013, H. Yin, K. Tang, Y. Gao, F. Klawonn, M. Lee, T. Weise, B. Li, and X. Yao, Eds., Berlin, I. D. Engineering and A. Learning, Eds. Heidelberg: Springer, 2013, pp. 611–618.

[57] J. Srinivasaraghavan and V. Allada, "Application of mahalanobis distance as a lean assessment metric," Int J Adv Manuf Technol, vol. 29, no. 11, pp. 1159–1168, August 2006.

[58] X.-Y. Li, Y. Wang, and W.-Z. Song, "Applications of k-local mst for topology control and broadcasting in wireless ad hoc networks," IEEE Transactions on Parallel and Distributed Systems, vol. 15, no. 12, pp. 1057–1069, December 2004.

[59] Z.-H. Zhou, "Ensemble learning," in Machine learning. Springer, 2021, pp. 181–210.

[60] A. Strehl and J. Ghosh, "Cluster ensembles — a knowledge reuse framework for combining multiple partitions," Journal of Machine Learning Research, vol. 3, pp. 583–617, 2002.

[61] X. Pennec, "Fréchet means in the space of statistics: a survey," HAL open science, 2020, working paper or preprint.

[62] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, June 2002.

[63] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017, pp. 1024–1034.

[64] W. Feng, J. Zhang, Y. Dong, Y. Han, H. Luan, Q. Xu, Q. Yang, E. Kharlamov, and J. Tang, "Graph random neural network for semi-supervised learning on graphs," in Advances in Neural Information Processing Systems 33 (NeurIPS 2020), 2020, pp. 6458–6468.

[65] J. Klicpera, S. Weißenberger, and S. Günnemann, "Graph diffusion convolution," in 8th International Conference on Learning Representations (ICLR 2020), 2020.

[66] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in Advances in Neural Information Processing Systems 31 (NeurIPS 2018), 2018, pp. 4800–4810.

[67] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 2623–2631.

[68] C. Zhang, J. Cui, and W. Liu, "Multilayer feature extraction of agcn on surface defect detection of steel plates," Computational intelligence and neuroscience, vol. 2022, no. 1, p. 2549683, 2022.