applied sciences

MDPI

Article

Anomaly Detection in Machining Centers Based on Graph Diffusion-Hierarchical Neighbor Aggregation Networks

Jiewen Huang and Ying Yang *

School of Computer, Electronics and Information, Guangxi University, Nanning 530004, China; ny000390@gmail.com
* Correspondence: yingy2004@126.com

Abstract: Inlight of the extensive utilization of automated machining centers, the operation and maintenance level and efficiency of machining centers require further enhancement. In our work, an anomaly detection model is proposed to detect the operation execution process by using the anomaly detection method of graph diffusion and graph neighbor hierarchical aggregation. In this paper, six machining center equipment states are defined and modeled, the monitoring sensors are referred to as nodes, and the connections between the sensors are represented as edges. First, we employed the graph diffusion model to enhance data quality within the sensor network model. Then, the node features were extracted using the hierarchical aggregation of neighboring nodes. Finally, after attentional connectivity, the ability of the model to learn global information was further improved. The performance of our model has been rigorously assessed using multiple experimental datasets and benchmarked against various anomaly detection techniques. The empirical findings unequivocally demonstrate the superior performance of our model, in terms of accuracy (96%) and F1 score (94), when compared to baseline models (MLP, GCN, GAT, GraphSAGE, GraphSAINT, GDC, and DiffusAL). The demonstrated effectiveness of the model underscores its versatility for a myriad of application prospects within the realm of manufacturing maintenance management.

Keywords: machining center; anomaly detection; graph neural network; diffusion model; attention mechanism

check for updates

Citation: Huang, J.; Yang, Y. Anomaly Detection in Machining Centers Based on Graph Diffusion-Hierarchical Neighbor Aggregation Networks. *Appl. Sci.* **2023**, *13*, 12914. https://doi.org/10.3390/app132312914

Academic Editor: Yutaka Ishibashi

Received: 30 October 2023
Revised: 28 November 2023
Accepted: 29 November 2023
Published: 2 December 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The technology of automated production lines has become widespread and extensively applied across various industries, including automotive manufacturing, electronic component production, and food processing. This technology has significantly enhanced production efficiency, reduced the utilization of human resources, and lowered costs. With the assistance of devices such as robotic arms and computer numerical control (CNC) machine tools, activities such as assembly, material processing, welding, and machining can be automated according to predefined programs, while human operators primarily engage in monitoring tasks. As the adoption of automation equipment increases in the manufacturing sector, competition among enterprises in terms of production efficiency intensifies. Thus, there is a growing importance of maintenance technology upgrades.

Predictive maintenance strategies have proven to effectively enhance production efficiency. In recent years, with the advancement of sustainable development reforms, maintenance strategies based on energy have gradually become mainstream [1–4]. With the advancement of intelligent manufacturing, the collection of production process parameters by enterprises has become increasingly sophisticated. The accumulation of a large amount of data provides a fertile ground for the development of data-driven diagnostic technologies, especially deep learning techniques. This, in turn, offers robust technical support for the implementation of the aforementioned maintenance strategies. Impressive results have been achieved [5,6]. However, early deep learning methods are usually applicable to regular data, such as images and sequences, and appear to be somewhat inadequate for graph data

with an irregular structure, where the correlation between the parameters measured by multiple sensors is often neglected. Graph representation learning [7], in particular, the advancement of the Graph Neural Network (GNN) method fills this gap [8–10]; the GNN method represented by the Graph Convolutional Network (GCN) and Graph Attention Network (GAT) can acquire the node characteristics along with their corresponding data and extract potential information from the data through the aggregation of neighboring node information and has become the mainstream in graph classification, node classification connection prediction, and other tasks and is widely used in chemistry, traffic, network security, and other fields [9,11–14]. However, the classical graph convolution method straightforwardly aggregates neighbor information, and the node features obtained are relatively shallow; what is more serious is that the simple aggregation method will bring more serious problems—the model is sensitive to noise. To improve the performance of the model on the test dataset, the presence of this factor impedes progress.

In order to overcome the challenges encountered by Graph Neural Network (GNN) methods in anomaly detection in processing centers, this study proposes improvements in the design from two aspects: dataset processing and aggregation methods. Sparse hierarchical neighbor aggregation is employed in neighbor aggregation and a skip connection mechanism is introduced. By focusing on measuring the importance of neighbor node representations at each aggregation layer, the model is better equipped to gain valuable data insights, enhance noise resistance, and ultimately improve the model's resistance to irrelevant information interference. This approach alleviates the training difficulty caused by the stacking of graph convolutional layers. Furthermore, in the data preprocessing stage, we introduce a data augmentation module based on the diffusion principle. This module optimizes workshop data facing issues, such as label sparsity, insufficient scale, and poor quality, thereby improving the efficiency of utilizing limited data. In conclusion, this study summarizes the development of anomaly detection techniques in processing centers and the challenges they face. Targeted improvements are proposed to enhance predictive accuracy and stability. The contributions can be summarized as follows:

1. In this study, a machining center anomaly detection model is proposed to detect the operation execution process of machining centers by using the anomaly detection method of graph diffusion and graph neighbor hierarchical aggregation (GraphDHNA). Possessing a high-precision capability for identifying processing anomalies, it can determine whether the machining quality is abnormal based on the state parameters. It provides a reliable anomaly detection tool for predictive and energy-oriented maintenance strategies, which can be widely applied in specific maintenance plans.
2. In this study, a data augmented module based on the diffusion principle is introduced in the graph neighbor aggregation network model to cope with the data quality problem.
3. In this study, the strategy of hierarchical aggregation is adopted in the process of neighbor aggregation; skip connections are added after each layer and attention connections are carried out in the last layer of the graph neighbor aggregation network, which fully considers the importance of the information of each step of neighbor aggregation, so that the features learned by the model are closer to the essence.
4. The model proposed in this study attains a higher level of accuracy (96%) and F1 score (94) when compared to both the classical GNN method and the presently popular GNN method. This performance improvement is observed across four datasets collected from real-world production environments.

The subsequent sections of this paper are organized as follows: Section 2 offers a comprehensive overview and summarizes the relevant literature in the domain of machining center fault detection. Section 3 provides a detailed description of the fault detection problem in the machining center, along with the relevant definitions. Section 4 outlines the methodology and modeling employed in this study. Section 5 conducts comprehensive experiments on the model introduced in this study and presents the findings of these

experiments, along with a thorough analysis. In conclusion, Section 6 offers a summary of the contributions made in this paper and outlines potential directions for future research.

2. Background and Related Work

The strategic processing of anomaly detection in machining centers can be categorized into three approaches: traditional periodic maintenance, predictive maintenance, and innovative energy-oriented maintenance. In terms of specific anomaly detection techniques, these can be further classified into traditional surface detection, traditional machine learning methods, and deep learning methods.

2.1. Maintenance Strategy

1. Regular maintenance (time-based maintenance): Regular maintenance is a basic maintenance strategy that involves conducting scheduled maintenance according to a predetermined maintenance plan. This approach is convenient to implement and is particularly suitable for components whose risk of failure increases with extended usage over time.
2. Predictive maintenance: Through real-time monitoring and data analysis, predictive maintenance anticipates the potential time of failure for equipment or systems, allowing for proactive maintenance measures to be taken before issues arise. The objective of this strategy is to minimize unplanned downtime, enhance equipment reliability and performance, and reduce maintenance costs.
3. Energy-oriented maintenance: Emphasizing the enhancement of equipment performance, reliability, and efficiency through the management and optimization of energy usage. This strategy incorporates energy efficiency into the scope of maintenance considerations. When certain devices and systems exhibit abnormal operating conditions, energy efficiency indicators can undergo significant changes. By monitoring energy efficiency, rational energy management measures can be implemented, leading to prolonged equipment lifespan, reduced energy wastage, and an overall improvement in equipment operational performance.

2.2. Methods of Detection

1. Traditional surface inspection: The conventional surface inspection involves the detection and identification of surface damage, substrate damage, internal defects, geometric dimensions, etc., through means such as visual inspection, dimension measurement, and nondestructive testing. These methods heavily rely on the proficiency of the personnel and the quality of their operations.
2. With the advancement of the informatization level in production workshops, the vast amount of production data has given rise to efficient diagnostic methods based on data analysis [15]. Early methods of data analysis involved examining the statistical features of historical data to identify early symptoms of faults. Adequate preparations were made to reduce the waste of manpower and spare part resources. However, early data analysis only explored the statistical characteristics of production data and failed to reveal the underlying features of fault occurrence, limiting the improvement of diagnostic accuracy. In the realm of machine learning, deep learning techniques involve the utilization of extensive datasets to train multilayered stacked neural network models for the acquisition of data features and are the most popular and effective data-driven class of anomaly detection methods today. Common deep learning anomaly detection models can be different according to their core neural network type; the more popular methods are as follows: convolutional neural network-based methods [5], recurrent neural network-based methods [6], Graph Neural Network-based methods, and so on [16]. They each possess their respective merits and limitations (as shown in Table 1), and enhanced diagnostic performance can be attained by selecting the suitable neural network architecture tailored to the particular context of the task data.

Table 1. Comparison of different neural networks.

Networks	Advantage	Disadvantage	Applications
MLP	Simple and intuitive; easy to understand and implement.	Sensitive to the structure and relationships of input data; limitations in processing sequence and graph structured data.	Used for basic classification and regression tasks with relatively simple structured data [17].
CNN	With translation invariance, it is suitable for processing data such as images, parameter sharing, and local connection help to reduce model parameters.	Sensitive to the size of the input data. Not suitable for modeling serial data.	Computer vision tasks such as image classification and target detection [18,19].
RNN	Capable of capturing timing information in sequence data and suitable for processing variable length input sequences.	Difficult to capture long-range dependencies and prone to gradient vanishing or exploding during the training process.	Natural language processing, time series prediction, and other tasks that require consideration of temporal relationships [20,21].
GNN	Ideal for working with graph-structured data; capable of capturing relationships between nodes.	Computational efficiency issues.	Social network analysis, molecular mapping prediction, numerous correlated signal inputs, and other tasks that require consideration of graph structure relationships [22–29].

As a means of addressing data-driven anomaly detection, neural network models are closely integrated with state monitoring, predictive maintenance, and energy-oriented maintenance. Mining potential information from data plays a crucial role in shaping maintenance strategies. Our work provides a model for industrial equipment maintenance management capable of handling complex network data with multiple signals, thereby enhancing the efficiency of maintenance management in machining centers. In the context of the evolution of neural network models, we have designed a novel Graph Neural Network model based on the characteristics of sensor data from machining centers. This model incorporates the advantages of diffusion models and attention mechanisms while improving the aggregation of neighboring nodes, thereby enhancing the performance of the diagnostic model. The key modules of our model include the following:

1. *Graph diffusion augment*: We designed the graph diffusion data augment module for multisensor variable graph data with the aim of obtaining better training samples. Because the edges and features of the graph are usually noisy, and graph diffusion smooths the neighborhoods on the graph, which can recover the regions where useful information exists in the noisy graph, it can, thus, be regarded as a denoising process [30], so that the new graph obtained by this processing will not lose the key information but, instead, the noisy information has been preliminarily filtered, and the newly added sample of the graphs has a higher information density in the original sample, so the graph diffusion augment module creates favorable conditions for model training.
2. *Neighborhood hierarchical aggregation*: The fundamental concept underlying Graph Neural Networks is to depict the target node as a consolidation of information gleaned from its neighboring nodes. This approach iteratively collects information from higher-order neighboring nodes, continually refining the representation of the target node throughout the process. In order to fully utilize higher-order relationships, it is often necessary to stack multiple graph convolutional layers together. However, previous research has identified that with an increase in the number of layers and model depth, a loss of gradient or excessive smoothing occurs, resulting in a marked decline in

performance [31,32]. In order to break through the limit of the number of layers, RestNet [33] designed a residual connection in the field of image processing; this strategy has achieved good results. Inspired by the idea of residual connection, we considered adopting a sparse aggregation method in the neighbor node information aggregation link. In this context, the process involves discreetly aggregating neighbor information at each order while simultaneously elevating the output of each layer, facilitating a direct transition to the output layer within the Graph Neural Network.

3. *Hop-attention*: Incorporating an attention mechanism enables a comprehensive evaluation of interrelationships among multiple variables, and the consideration of contextual information plays a big role in natural language processing [34]. The neighbor aggregation link in each layer output, after the skip connection, directly to the final layer of the graph neighbor aggregation network, where these are inputs for the attention connection; because each input represents the node representation of a certain hop neighbor, it is called the hop-attention layer; the mechanism comprehensively considers the weight of each layer on the final result in order to highlight the most important part of the information of the neighbors of each order. The immunity of the model to noise is improved.

3. Preparation

On the electromechanical integration equipment of machining centers, numerous sensors are installed. In this study, taking the example of the dual-swivel five-axis machining center manufactured by KeDe Numerical Control Co., Ltd. (Dalian, China), the machine tool and processing head collectively possess three linear axes (X, Y, and Z) and two rotary axes (A and B). Each axis is equipped with sensors for temperature, pressure, flow, current, voltage, load, speed, torque, position (grating ruler), etc., which come preinstalled from the factory. There exists a topological relationship in terms of communication between the sensors, logical layout, and the physical significance of the monitored quantities. Considering the characteristics and data features of fault detection tasks in machining centers based on data from multiple sensors, we define the construction method of a multisensor association graph and the transformation rules from sensor entities to abstract networks.

3.1. Definition of Sensors Network Graph

The raw data comprise machining information for four components of a specific engine model. They are categorized into four subsets based on the different types of components, with each subset representing the operational data collected during the machining of a specific type of product. Each execution of a machining operation results in a set of data records from the sensors in the processing center. Each sensor within a data record serves as a node, with its measured values serving as node features, and a fixed scanning frequency determining the node features. Abstracting the operational information into a graph transforms the anomaly detection task into a graph classification problem. In this task, six types are categorized according to the actual situation, i.e., five anomaly types depending on the region where the anomaly occurs and an anomaly-free type. Specifically, we define an undirected graph $G = (V, E)$, which has a set of nodes V, representing individual sensors, and a set of edges E, representing the connecting edges between the sensors. We use $N = |V|$ to denote the number of nodes and $A \in R^{N \times N}$ to denote the adjacency matrix.

3.2. Connection between Nodes

As is well known, a graph is composed of entities and their connections, where each entity is represented as a node in the graph, and the relationships between two entities correspond to edges. Based on the similarity or dissimilarity of node attributes, graphs can be classified as homogeneous or heterogeneous; based on the direction of edges, graphs can be categorized as directed or undirected. Considering the data characteristics of the processing center, we construct the task data as an undirected homogeneous graph. Each sensor

(or the measurement values from a sensor) corresponds to a node in the undirected graph. During the processing operations, sensors collect data at a specified scanning frequency, sequentially serving as node features while the correlations between measurement values serve as edges. These correlations encompass both physical connections and connections in the actual sensor environment, as well as those determined based on industry experts' and maintenance personnel's experiences.

When applying Graph Neural Networks to the field of anomaly detection in processing centers, the degree of freedom in the design of sensor network configurations serves as both an opportunity and a challenge. In particular, the significance of empirical knowledge in this domain underscores the importance of limited labels. Effectively utilizing limited data and exploring/expanding new node connections are crucial for achieving the ultimate accuracy in anomaly detection.

4. Methodology

4.1. General Model Framework

The schematic diagram of the model we designed is illustrated in Figure 1. Broadly, the model can be divided into two main components: the first part is the graph diffusion augmentation section and the second part comprises the neighborhood information aggregation layers of the Graph Neural Network (GNN), along with the attention layers for each neighborhood aggregation information, and the readout layer that consolidates the overall representation of the computation graph. Initially, the operational parameters of the automated production workshop's processing center are structured into a graph dataset, and the original graph dataset undergoes graph diffusion augmentation to generate additional expanded data. These data then enter the GNN for the neighborhood information aggregation section, where, through multiple iterations and multistep neighborhood information aggregation, preliminary node representations are obtained. Subsequently, after attention operations and the readout layer, the final graph feature representation is acquired. The classifier then produces probability outputs.

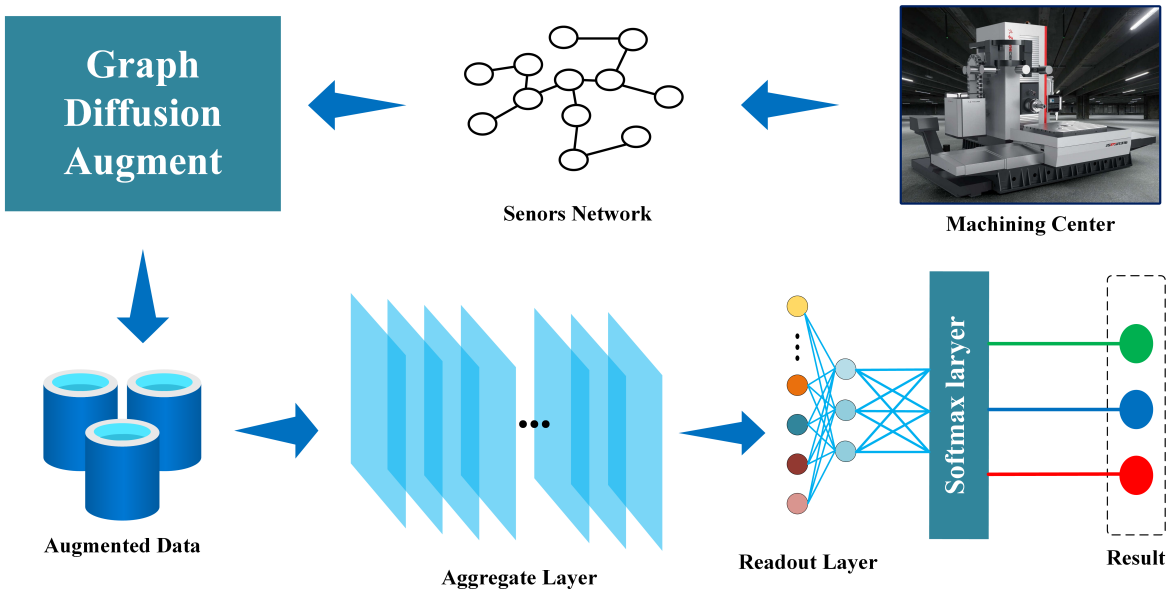


Figure 1. The overall framework of GraphDHNA model. The original data are constructed into an undirected graph, which is enhanced by diffusion and inputted into the neighborhood information aggregation layer, in which hierarchical aggregation and hop-attention allocation are carried out, and then the diagnosis results are obtained through the readout layer and classifier.

4.2. Diffusion Augmentation Module

In summary of the foregoing, the Graph Neural Network (GNN) method emerges as an excellent choice for processing center anomaly detection. However, typical GNN models often require a substantial amount of training data and a considerable labeling rate. In practical scenarios, we encountered limitations in the availability of fault data, with the quantity of unlabeled data often far exceeding that of labeled data. While theoretically, these issues could be alleviated by increasing data collection efforts, data augmentation and labeling often demand several times the manpower and resources. This is particularly true as many application domains require high levels of expertise and experience from practitioners for labeling, further constraining the feasibility of large-scale data augmentation and labeling.

Therefore, rather than directly augmenting the original data, it is more prudent to explore methods to unearth latent information within existing data and effectively leverage limited data resources. In the fields of image and natural language processing, the use of generative models to address data scarcity has been proven to be an effective strategy [35,36]. Generative models, such as diffusion models, can produce training samples that closely resemble the original dataset. Although these generated samples are artificial, they still exhibit features related to real samples, effectively enlarging the scale and improving the quality of the training dataset.

Diffusion models, currently a popular class of generative models, employ a differentiable inverse process and gradually transform the target distribution into the initial distribution, exhibiting strong generative capabilities. Leveraging the concept of diffusion models, nodes and edges on a diffusion graph are diffused, extending the source data and generating original data that may not exist in all samples, thereby significantly overcoming the limitations imposed by sample size.

In processing center anomaly detection, the data environment often presents challenges due to the prevalence of normal equipment operation compared to equipment failures. Additionally, the correlations between multiple sensors are established based on production experience and surface-level connections, not fully encompassing the connections between data. Consequently, such data samples often suffer from issues of insufficient labeling. The principles of diffusion can offer significant advantages in such data environments. Moreover, existing GNN methods typically only utilize the direct neighbors of each node, assigning them equal weights during information aggregation [11]. However, the connections between entities are not always direct, and the underlying features behind the graph are often more complex. The constructed graphs often only capture partial information. Applying diffusion models in anomaly detection maximizes the value of limited data, enhances the compatibility of the model across varying data qualities, and plays a crucial role in the widespread deployment of models in practical production scenarios. In the foreseeable future, diffusion models are expected to frequently emerge in anomaly detection across various industrial domains, particularly in fields where the quality of raw data is not high, making it a primary stronghold for the application of diffusion models.

The fundamental concept underlying the graph diffusion model is that it facilitates the smoothing of neighborhoods on the graph, serving as a denoising filter akin to a Gaussian filter applied to an image. This facilitates the model in acquiring vital information pertaining to the graph, as features and edges in graphs within real-world environments tend to exhibit inherent noise. Prior research has similarly underscored the robustness of this concept [37–39]. The studies affirm that graph diffusion-based smoothing can effectively extract valuable neighborhood information from noisy graphs. Graph diffusion is performed as follows. First, we can define the parameter matrix of the new graph $\mathcal{G}$ generated by the diffusion of the undirected graph G as follows:

$$\mathcal{G} = \sum_{k=0}^{\infty} \theta_k T^k, \tag{1}$$

where $k \in \mathcal{R}$, the transition matrix is denoted as T, and θ_k is the weighted coefficient. The principle of selecting T and θ is to ensure the series convergence of Equation (1). The related

work has demonstrated that the choice of coefficients defined by Personalized PageRank (PPR) and the heat kernel yields strong experimental performance [40–44]. PPR constructs the transfer matrix T_{rw} by walking randomly, which can be written as Equation (2). The random walk starts from the starting node, selects the next node according to the probability distribution, and then continues to walk until the specified number of walking steps or the termination condition is reached. In each step, the transition probability of the node can be adjusted according to the structure of the graph or personalized preferences. $\theta_k^{PPR} = \alpha(1 \times \alpha)^k$. $\alpha \in (0, 1)$ represents the propagation probability while the heat kernel is defined by $T = T_{rw}, \theta_k^{HK} = e^{-k}\frac{t^k}{k!}$, where $t \in [1, 10]$ is the diffusion time. And, as in [45], the symmetrical version of T_{sym} is selected as the transition matrix, which can be written as Equation (3).

$$T_{rw} = AD^{-1} \tag{2}$$

$$T_{sym} = D^{-1/2}AD^{-1/2} \tag{3}$$

$$T_{gat} = softmax(A + I) \tag{4}$$

The matrix I is the unit matrix with dimensions matching those of the graph G, and the matrix D represents the symmetric matrix of the node degrees within the graph G, so that

$$D_{ii} = \sum_{j=1}^{N} A_{ij}. \tag{5}$$

The previous work found that T_{sym} outperforms T_{rw} in most cases [45], and this advantage relies on the insensitivity when introducing self-loops of the node's own information, although self-loops introduce the dissemination of the node's proprietary information, in the GCN approach, because the weights of the self-loop are usually learned and it only involves the node's own information and not interactions with other nodes. Therefore, the self-loop is a relatively small change to the model. In the case of self-loops, we observe that the GAT model outperforms the GCN approach, as reported in prior work [45]; the self-loop can be viewed as a regularization mechanism. It imposes a constraint on the learning of the attention weights by ensuring that the node's own information is taken into account in the attention computation. This measure can aid in mitigating the overfitting of the model during the training process. Therefore, in this study, we used the transfer matrix T_{gat} for the GAT method, which can be expressed as Equation (4), and $\theta_k = \alpha(1 - \alpha)^k$, $\alpha \in (0, 1)$ for the PPR method. So, one can write Equation (1) in the following form:

$$\mathcal{G} = \sum_{k=0}^{\infty} \alpha(1 - \alpha)^k \cdot softmaxt(A + I)^k. \tag{6}$$

4.3. Neighborhood Information Aggregate

The main task performed by the graph convolution module is the feature extraction of the input graph data, mapping the nodes to vectors in the representation space, and then unifying the nodes into graph-level representations for subsequent classification. The module consists of a hierarchical neighbor information aggregation layer and a hop-attention connectivity layer for each order of neighbor aggregation information, as shown in Figure 2.

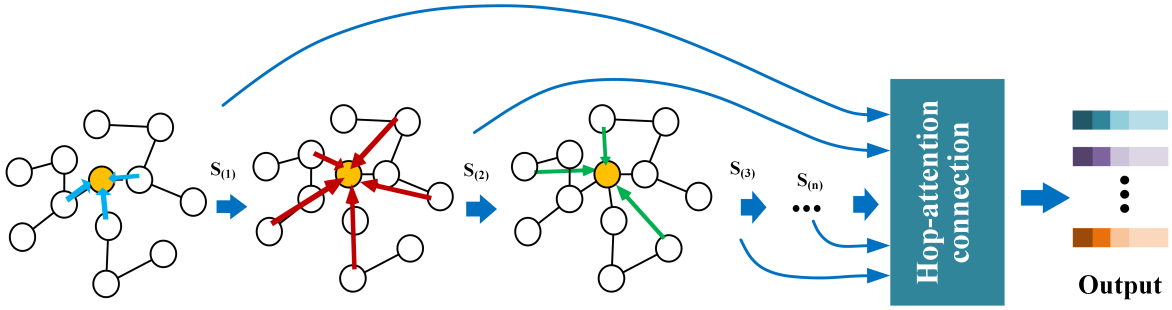


Figure 2. Hierarchical aggregation of neighbor nodes. Multiorder neighbors of a node are hierarchically aggregated for information and then attentively connected to obtain the target node representation.

4.3.1. Aggregate Layer

Early approaches to graph convolution sought to port two-dimensional convolutional networks onto irregular graphs [11], which naturally applied convolutions like in the image domain by representing solid networks onto planar matrices via time–frequency transformations; this category of approach is centered on acquiring knowledge regarding the structural characteristics of the network. Starting from GAT, the computation of node weights began to transition from being primarily based on the structural attributes of the network to being primarily reliant on the feature representations of individual nodes, but these models need to load the node features of the whole network when processing, which causes a certain training threshold and hinders the application of the models on large-scale networks.

Hamilton et al. proposed Graph Sample and Aggregated Embeddings (GraphSAGE) [46]. Unlike previous models that consider all neighboring nodes, Graph Sampling Aggregation Networks perform a random sampling of neighboring nodes such that each node's neighboring nodes are less than a given number of samples. However, GraphSAGE may introduce sample bias when nodes' neighbors have different degrees. For example, in a highly degree-centered graph, nodes with higher degrees may be selected more frequently in the sampling while nodes with lower degrees are ignored, causing the model to favor highly connected nodes in the node representation. Therefore, we considered the full sampling of neighboring nodes of a specific order and subsequently assigned the weights of the neighbor information of each order, which can avoid sampling bias without causing a dramatic increase in computational cost. In our designed aggregation layer, its job is to perform the multi-neighbor information aggregation of nodes; through the successive stacking of multiple aggregation layers, this process continually extends the aggregation from lower-order neighbors to higher-order neighbors, in addition to the information propagation between layers. Unlike GraphSAGE's gradually expanding sampling graph, we adopted the strategy of aggregating only the specified order of neighbor information at a time, i.e., when aggregating l-order neighbor information, we did not repeat the aggregation of $l-1$-order neighbor information (since the $l-1$-order neighbor information has already been accessed in the previous layer). In addition, each layer introduces a skip connection that skips directly to the last layer to correct the weights of the final aggregation result. The primary objective is to alleviate potential overfitting resulting from the increase in the number of convolutional layers and to diminish the influence of extraneous information. We define the form of the node update, as in (7):

$$s^{(l+1)} = s^{(l)} + \frac{\sum_{i \in |N_{(l)}|} s_i^{(l)}}{|N_{(l)}|}. \tag{7}$$

In the provided expression, s represents the node embedding of the current layer, l corresponds to the current layer number, and $N_{(l)}$ signifies the set of nodes that are

neighbors to node "n" at the lth order. Furthermore, $|N_{(l)}|$ quantifies the total count of lth order neighbor nodes.

4.3.2. Hop-Attention

Neighbors of nodes of each order in the previous aggregation level indicate that the outputs are connected via skips, and the computation of weight allocation is carried out at this level, and the attention mechanism takes into account all the inputs through clever manipulation, and after the attention scores, the most important part of the information is judged. The outputs $s_1, s_2, s_3 \ldots s_n$ of each aggregation level are able to fully consider the importance of different orders of neighbor information during the multihop neighbor aggregation process through the self-attention mechanism at the hopping level, which reduces the influence of the noisy information aggregated during the aggregation of the neighbor information on the final model determination. On the other hand, the hop-attention connection has a lower risk of overfitting than a simple summation or mean fusion, which gives the same weight to each hop neighborhood and will over-learn useless features in the training data, achieving an accuracy on the training dataset that is difficult to achieve on the test set. The specific approach of the attention mechanism is to take multiple inputs with correlations, map them into *Query*, *Key*, and *Value*, and then compute the respective attention scores in order to reassign weights to each input [34]. The weight matrix W is first used to map s into Q, K, and V.

$$Q^i = W^q s_i \tag{8}$$

$$K^i = W^k s_i \tag{9}$$

$$V^i = W^v s_i, \tag{10}$$

where W^q, W^k, W^v are weight matrices that map the outputs of each aggregation layer to Q, K, V.

$$\alpha_{i,j} = \frac{Q^i \cdot K^j}{\sqrt{d}}, \tag{11}$$

where α is the attention score, representing the correlation or similarity between q and k. It is a probability distribution used to determine which v is weighted when calculating the output. A higher attention weight means that elements more related to q will receive more weight while a lower weight means that less-related elements will receive less weight. Here, we used scaled dot product attention, where the denominator d represents the dimension of q, which can limit the size of the dot product of q and k in the numerator, ensuring that the distribution of the attention scores does not become too scattered due to the input dimension.

$$\hat{\alpha_{i,j}} = \frac{\exp(\alpha_{i,j})}{\sum_j \exp(\alpha_{i,j})} \tag{12}$$

$\hat{\alpha}$ is a softmax operation on α, which normalizes the spatial correlation of the outputs of each layer.

$$\hat{s_i} = \sum_j \hat{\alpha_{i,j}} V^j \tag{13}$$

The primary function of the "V" operation is to generate the ultimate output, denoted as $\hat{s}$, based on the calculated attention weights, which allows the mechanism to select and combine values in a targeted manner based on the correlation between Q and K; the modification allows the model to more effectively concentrate on pertinent information within the input data, and concentrating the features learned by the model on the key parts

to improve the performance of the task, thus far we obtain a node vector representation of the processed nodes.

4.3.3. Readout Layer

After neighbor information aggregation and attention connection, we obtain a node-level vector representation representing each sensor node, and in the classification of skips, one job corresponds to one graph, and a graph-level representation is needed to classify it. The function of the readout layer is to unify the node-level vector representations obtained in the previous steps to obtain a unified vector representation at the graph level and to classify them in the representation space, and the readout layer in this study is in the form of a fully connected layer readout.

5. Experiments

5.1. Data Description

This study used four real-world machining center case datasets, based on raw data provided by a certain internal combustion engine manufacturer, and divided them into four datasets according to the types of mechanical components processed, namely the machining process data of gears, crankshafts, end caps, and pistons. Their detailed information is shown in Table 2.

Table 2. Data description.

Datasets	Graphs	Nodes	Edges	Features
S06_gear	102	5148	16,329	1325
S06_cylinder	192	9331	25,746	1395
S06_shaft	168	8435	43,924	1665
S06_piston	183	9726	51,123	1495

5.2. Experimental Setups

To substantiate the efficacy of GraphDHNA, we set up four groups of co-benchmark control models, namely multilayer perceptron (MLP) for non-GNN methods, GCN [11] and GAT [47] for base GNN, GraphSAGE and GraphSAINT [48] for neighbor node sampling, and GDC [45] and DiffusAL [30] for graph diffusion methods. The metrics commonly employed in the classification tasks, such as classification accuracy in Equation (14) and $F1$ scores in Equation (15), are used as the evaluation metrics.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

$$F1 = 2 \cdot \frac{(TP/TP + FP) \cdot (TP/TP + FN)}{TP/(TP + FP) + TP/(TP + FP)} \tag{15}$$

In the context of this study, TP, TN, FP, FN represent the counts of true positives, true negatives, false positives, and false negatives, respectively.

In this section, all experiments were conducted on a server equipped with 40 CPU cores, DDR4 512 GB, and eight Nvidia V100 GPUs, with two Intel XeonGold 6230 2.1 GHz CPUs, using Python version 3.8.0 and PyTorch version 2.0.1. And, the number of aggregation layers was four; we split each dataset, where 80% was used for model training, 20% for testing, and 20% for validation. All models were trained with 200 epochs, where the training loss variation of GraphDHNA on the four datasets is shown in Figure 3. More experimental results can be found in Figures A1–A12.

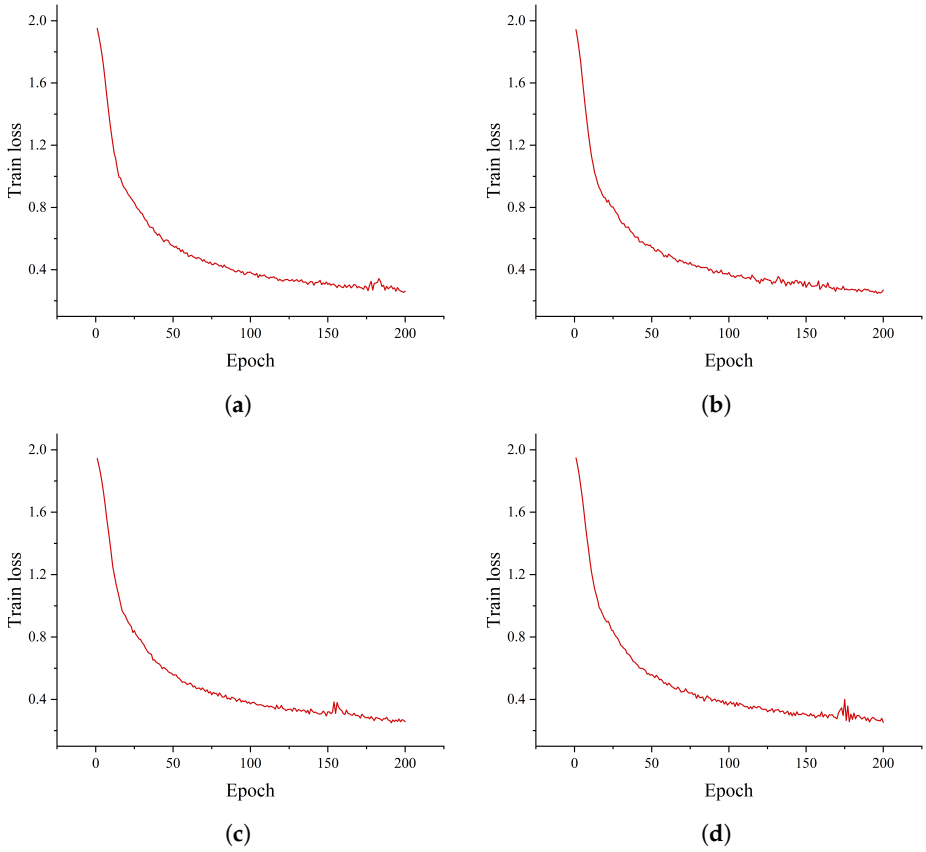


Figure 3. GraphDHNA's loss variation on four training datasets. (**a**) Training on S06_gear; (**b**) training on S06_cylinder; (**c**) training on S06_shaft; (**d**) training on S06_piston.

5.3. Experimental Results

We tested the multilayer perceptron (MLP), the classical Graph Neural Network models, GCN and GAT, and the currently popular GDC for comparison and tuned the reference model as best as possible according to the optimal experimental setups provided by references, trained and tested on four datasets, and saved the optimal model parameters after 200 rounds of training; on the test set, respectively, 10 tests were performed and then averaged. The accuracy and *F*1 scores of each model are shown in Table 3.

Table 3. Experimental results.

Metrics	Methods	S06_gear	S06_cylinder	S06_shaft	S06_piston
Acc	MLP	82.23 ± 0.52	82.99 ± 0.38	83.52 ± 0.42	84.09 ± 0.33
	GCN	90.29 ± 0.72	90.09 ± 0.29	90.52 ± 0.28	91.11 ± 0.11
	GAT	90.79 ± 0.12	91.50 ± 0.45	91.93 ± 0.19	92.35 ± 0.07
	GraphSAGE	91.28 ± 0.20	91.58 ± 0.34	92.01 ± 0.43	92.63 ± 0.23
	GraphSAINT	91.35 ± 0.13	92.15 ± 0.18	91.98 ± 0.67	92.57 ± 0.64
	GDC	92.86 ± 0.12	92.60 ± 0.46	93.03 ± 0.51	93.63 ± 0.38
	DiffusAL	92.95 ± 0.32	92.64 ± 0.33	93.07 ± 0.19	93.69 ± 0.26
	GraphDHNA	94.99 ± 0.25	94.83 ± 0.31	94.32 ± 0.18	96.13 ± 0.24
F1	MLP	80.29 ± 0.63	80.09 ± 0.41	81.52 ± 0.25	82.11 ± 0.24
	GCN	86.67 ± 0.72	87.82 ± 0.67	86.36 ± 0.74	87.32 ± 0.53
	GAT	87.76 ± 0.46	87.91 ± 0.54	86.45 ± 0.33	88.41 ± 0.51
	GraphSAGE	88.89 ± 0.60	89.04 ± 0.48	87.58 ± 0.46	89.54 ± 0.46
	GraphSAINT	87.07 ± 0.39	89.22 ± 0.53	88.76 ± 0.64	89.72 ± 0.44
	GDC	89.54 ± 0.29	90.69 ± 0.19	89.23 ± 0.51	90.19 ± 0.49
	DiffusAL	90.35 ± 0.48	91.23 ± 0.35	90.04 ± 0.61	91.39 ± 0.42
	GraphDHNA	92.69 ± 0.36	92.13 ± 0.24	92.41 ± 0.16	94.38 ± 0.21

As seen from the experimental results in Table 3, the simple MLP method performs poorly, reaching only 84% classification accuracy; the classical GNN methods, such as GCN and GAT, intersect MLP with a sizable improvement, reaching the 90% threshold; the graph sampling method further obtains an improvement of about 10%, reaching 91–92%. The graph diffusion GDC and DiffusionAL models achieved a more significant advantage among all the baseline models, reaching a classification accuracy of 93%. GraphDHNA achieved the highest scores in both classification accuracy and *F*1 scores, reaching 96% and 94, which surfaces the superiority of GraphDHNA in dealing with machining center anomaly detection tasks.

5.4. Ablation

In order to target and study the effects of the graph diffusion augmentation module (GDA), graph neighbor hierarchical aggregation module (HA), and hop-attention connection module (HAT) on the performance of the GraphDHNA model, we conducted ablation experiments on the GraphDHNA model by setting up the experimental groups that removed the graph diffusion data augmentation module, the experimental group that removed the attentional connection, and the experimental group that removed the hierarchical neighbor aggregation module, which needed to be emphasized so that these experimental groups kept the settings consistent with GraphDHNA except for the module for targeted changes, which ensured the singularity of the experimental variables. The test accuracies and *F*1 scores for the three sub-models and GraphDHNA are shown in Table 4. As shown by the results of the ablation experiments, each module of GraphDHNA plays an active role, and removing any one of the modules causes a performance degradation (e.g., a two percentage point drop in classification accuracy), and removing two modules causes a larger performance degradation (e.g., a three percentage point drop in classification accuracy).

Table 4. Ablation result ("✓" means including the module, "×" means module is not included).

Metrics	GDA	HA	HAT	S06_gear	S06_cylinder	S06_shaft	S06_piston
Acc	×	✓	×	90.65 ± 0.05	89.76 ± 0.05	92.13 ± 0.06	93.17 ± 0.11
	✓	✓	×	91.34 ± 0.05	92.58 ± 0.04	90.89 ± 0.04	94.21 ± 0.05
	×	✓	✓	91.34 ± 0.05	92.58 ± 0.04	90.89 ± 0.04	94.21 ± 0.05
	✓	×	✓	91.52 ± 0.05	92.02 ± 0.04	91.29 ± 0.05	92.51 ± 0.06
	✓	✓	✓	94.99 ± 0.25	94.83 ± 0.31	94.32 ± 0.18	96.13 ± 0.24
F1	×	✓	×	89.97 ± 0.11	90.58 ± 0.07	91.71 ± 0.13	91.88 ± 0.05
	✓	✓	×	90.16 ± 0.05	89.63 ± 0.05	90.55 ± 0.06	92.38 ± 0.05
	×	✓	✓	91.39 ± 0.11	90.87 ± 0.09	91.02 ± 0.13	92.13 ± 0.08
	✓	×	✓	91.34 ± 0.05	92.58 ± 0.04	90.89 ± 0.04	94.21 ± 0.05
	✓	✓	✓	92.69 ± 0.36	92.13 ± 0.24	92.41 ± 0.16	94.38 ± 0.21

6. Conclusions

In summary, this article proposes a graph diffusion-enhanced hierarchical aggregation attention network model for the anomaly detection of machining centers. Firstly, we constructed an undirected graph of multisensor data from the production process. The model diffuses and amplifies the underlying graph data to tackle quality issues within the original dataset and to comprehensively unearth the latent insights within the data. The neighbor aggregation network layer of this model adopts hierarchical aggregation, followed by hop attention connectivity, which can effectively capture key features on the graph and reduce the risk of overfitting. It achieves a classification accuracy of 96% and an F1 score of 94 on the test dataset, surpassing other comparative models. This means that our model demonstrates the capability to identify a greater number of anomalous samples while concurrently ensuring a relatively low rate of misjudgments. This holds particular significance for cost control in the realm of maintenance management. In summary, we contributed an endeavor involving the application of a graph diffusion neighbor aggre-

gation model to anomaly detection in machining centers. Nevertheless, it is imperative to acknowledge the limitations inherent in our work, such as the relatively small scale of the dataset and the exclusion of spatial, temporal, and other contextual information from consideration. In future work, we will attempt to explore the potential structural features of work process data on larger graphs and consider more information, including location and timing.

Author Contributions: Conceptualization, J.H. and Y.Y.; data curation, J.H.; formal analysis, J.H.; funding acquisition, Y.Y.; investigation, J.H. and Y.Y.; methodology, J.H. and Y.Y.; project administration, Y.Y.; resources, Y.Y.; software, J.H.; supervision, Y.Y.; validation, J.H.; visualization, J.H.; writing—original draft, J.H.; writing—review and editing, J.H. and Y.Y. All authors have read and agreed to the published version of this manuscript.

Funding: This paper was supported by Guangxi Science and Technology Department (Guikejizi [No. 2021175]), Grant number: AD21076002.

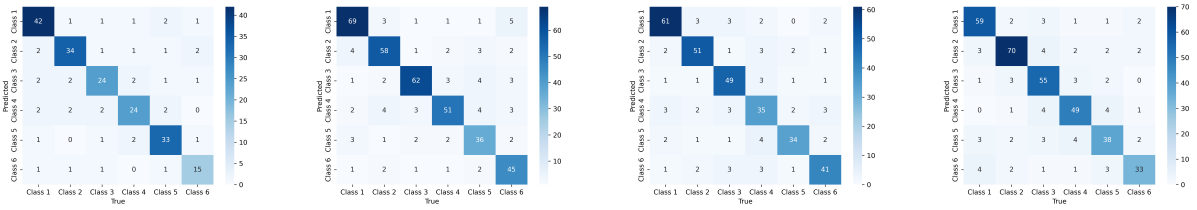
Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The machining center data used in this study were obtained from Guangxi Yuchai Machinery Group Co. The availability of these data is restricted and these data are used under license for this study and are not made publicly available. The authors may make these data available with the permission of Yuchai Engine Co. All data processing and analysis comply with relevant data protection and privacy laws. No personal or individual data were used in this study.

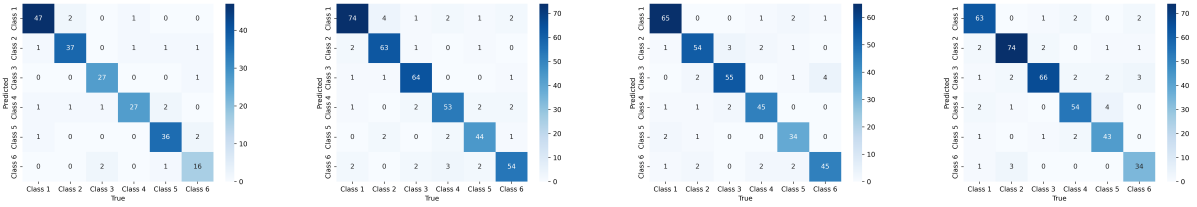
Conflicts of Interest: The authors declare no conflict of interest.

Appendix A



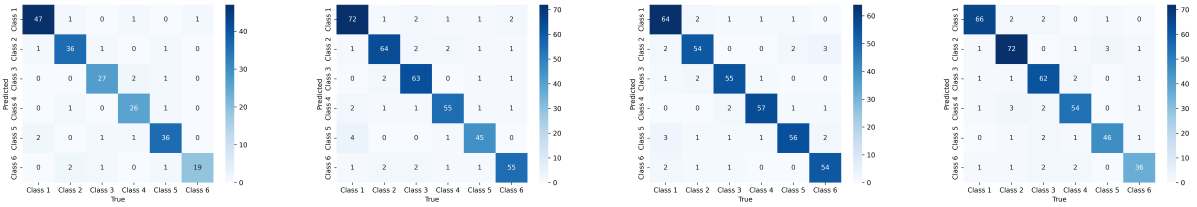
(**a**) (**b**) (**c**) (**d**)

Figure A1. Confusion matrix heat map of MLP's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



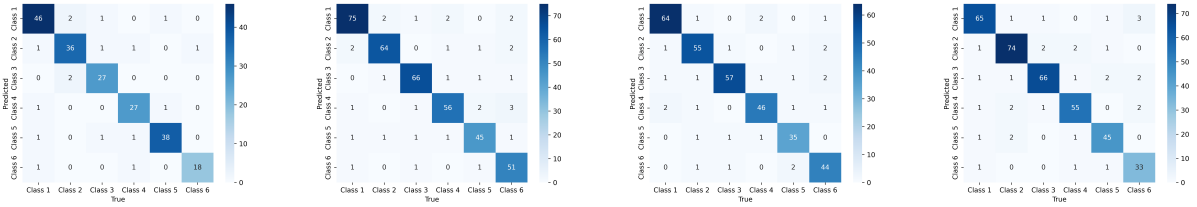
(**a**) (**b**) (**c**) (**d**)

Figure A2. Confusion matrix heat map of GCN's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



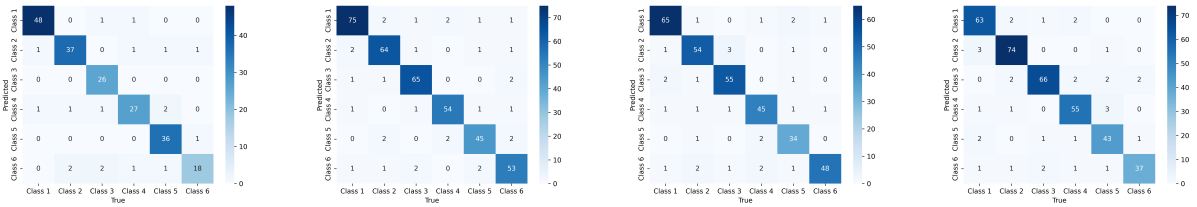
(**a**) (**b**) (**c**) (**d**)

Figure A3. Confusion matrix heat map of GAT's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



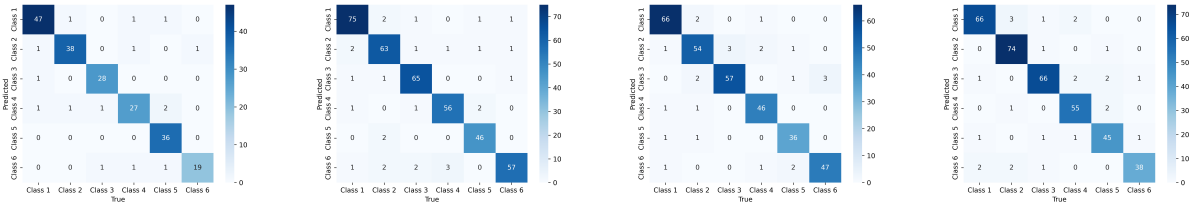
(**a**) (**b**) (**c**) (**d**)

Figure A4. Confusion matrix heat map of GraphSAGE's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



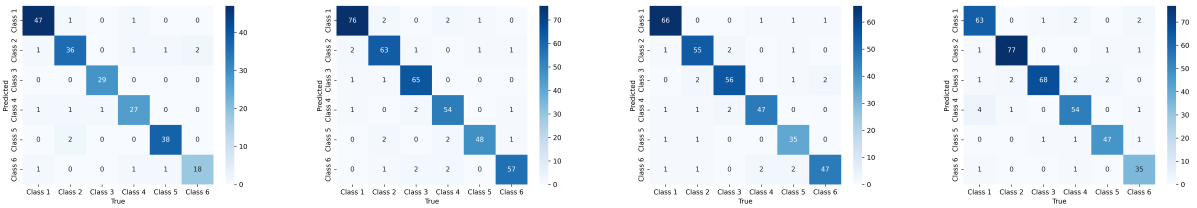
(**a**) (**b**) (**c**) (**d**)

Figure A5. Confusion matrix heat map of GraphSAINT's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



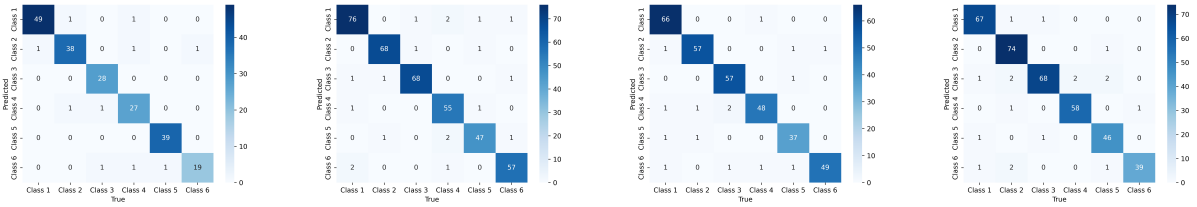
(**a**) (**b**) (**c**) (**d**)

Figure A6. Confusion matrix heat map of GDC's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



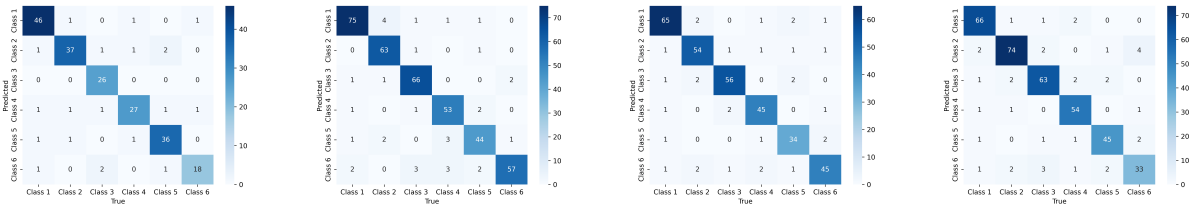
(**a**) (**b**) (**c**) (**d**)

Figure A7. Confusion matrix heat map of DiffusAL's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



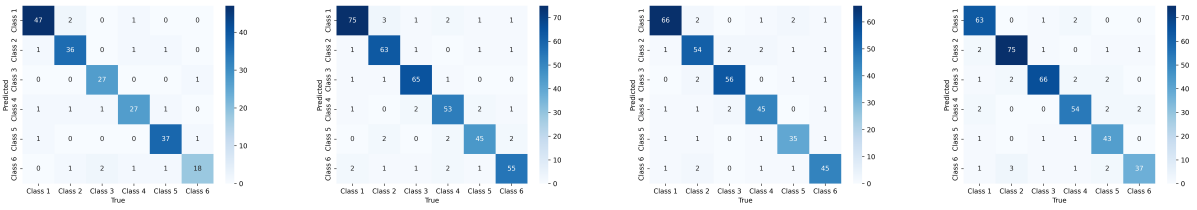
(**a**) (**b**) (**c**) (**d**)

Figure A8. Confusion matrix heat map of GraphDHNA's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



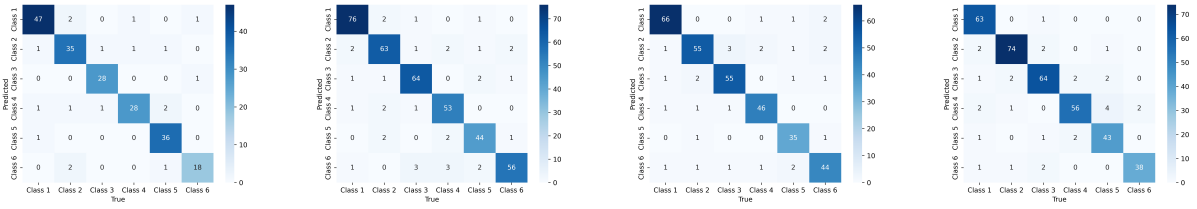
(**a**) (**b**) (**c**) (**d**)

Figure A9. Confusion matrix heat map of Ablation1's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



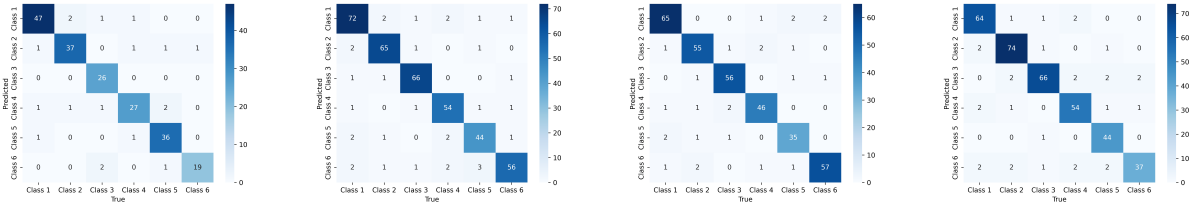
(**a**) (**b**) (**c**) (**d**)

Figure A10. Confusion matrix heat map of Ablation2's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



(**a**) (**b**) (**c**) (**d**)

Figure A11. Confusion matrix heat map of Ablation3's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.



(**a**) (**b**) (**c**) (**d**)

Figure A12. Confusion matrix heat map of Ablation4's test results on four datasets. (**a**) Test on S06_gear; (**b**) test on S06_cylinder; (**c**) test on S06_shaft; (**d**) test on S06_piston.

References

1. Zhou, B.; Yi, Q. An energy-oriented maintenance policy under energy and quality constraints for a multielement-dependent degradation batch production system. *J. Manuf. Syst.* **2021**, *59*, 631–645. [CrossRef]
2. Orošnjak, M.; Brkljač, N.; Šević, D.; Čavić, M.; Oros, D.; Penčić, M. From predictive to energy-based maintenance paradigm: Achieving cleaner production through functional-productiveness. *J. Clean. Prod.* **2023**, *408*, 137177. [CrossRef]
3. Orošnjak, M.; Jocanović, M.; Čavić, M.; Karanović, V.; Penčić, M. Industrial maintenance 4(.0) horizon Europe: Consequences of the iron curtain and energy-based maintenance. *J. Clean. Prod.* **2021**, *314*, 128034. [CrossRef]
4. Xia, T.; Xi, L.; Du, S.; Xiao, L.; Pan, E. Energy-oriented maintenance decision-making for sustainable manufacturing based on energy saving window. *J. Manuf. Sci. Eng.* **2018**, *140*, 051001. [CrossRef]
5. Eren, L.; Ince, T.; Kiranyaz, S. A generic intelligent bearing fault diagnosis system using compact adaptive 1D CNN classifier. *J. Signal Process. Syst.* **2019**, *91*, 179–189. [CrossRef]
6. Zhu, J.; Jiang, Q.; Shen, Y.; Qian, C.; Xu, F.; Zhu, Q. Application of recurrent neural network to mechanical fault diagnosis: A review. *J. Mech. Sci. Technol.* **2022**, *36*, 527–542. [CrossRef]
7. Hamilton, W.L. *Graph Representation Learning*; Morgan & Claypool Publishers: San Rafael, CA, USA, 2020.
8. Bronstein, M.M.; Bruna, J.; LeCun, Y.; Szlam, A.; Vandergheynst, P. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Process. Mag.* **2017**, *34*, 18–42. [CrossRef]
9. Gilmer, J.; Schoenholz, S.S.; Riley, P.F.; Vinyals, O.; Dahl, G.E. Neural message passing for quantum chemistry. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 1263–1272.
10. Battaglia, P.W.; Hamrick, J.B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. Relational inductive biases, deep learning, and graph networks. *arXiv* **2018**, arXiv:1806.01261.
11. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
12. Zhang, M.; Chen, Y. Link prediction based on graph neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018.
13. Zhao, L.; Song, Y.; Zhang, C.; Liu, Y.; Wang, P.; Lin, T.; Deng, M.; Li, H. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3848–3858. [CrossRef]
14. Lee, J.B.; Rossi, R.; Kong, X. Graph classification using structural attention. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1666–1674.
15. Lei, Y.; Yang, B.; Jiang, X.; Jia, F.; Li, N.; Nandi, A.K. Applications of machine learning to machine fault diagnosis: A review and roadmap. *Mech. Syst. Signal Process.* **2020**, *138*, 106587. [CrossRef]
16. Chen, Z.; Xu, J.; Alippi, C.; Ding, S.X.; Shardt, Y.; Peng, T.; Yang, C. Graph neural network-based fault diagnosis: A review. *arXiv* **2021**, arXiv:2111.08185.
17. Yu, J.; Zhang, Y. Challenges and opportunities of deep learning-based process fault detection and diagnosis: A review. *Neural Comput. Appl.* **2023**, *35*, 211–252. [CrossRef]
18. Zhang, Z.; Wang, H.; Xu, F.; Jin, Y.Q. Complex-valued convolutional neural network and its application in polarimetric SAR image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 7177–7188. [CrossRef]
19. Ajad, A.; Saini, T.; Niranjan, K.M. CV-CXR: A Method for Classification and Visualisation of Covid-19 virus using CNN and Heatmap. In Proceedings of the 2023 5th International Conference on Recent Advances in Information Technology (RAIT), Dhanbad, India, 3–5 March 2023; pp. 1–6.
20. Toharudin, T.; Pontoh, R.S.; Caraka, R.E.; Zahroh, S.; Lee, Y.; Chen, R.C. Employing long short-term memory and Facebook prophet model in air temperature forecasting. *Commun. Stat. Simul. Comput.* **2023**, *52*, 279–290. [CrossRef]
21. Farah, S.; Humaira, N.; Aneela, Z.; Steffen, E. Short-term multi-hour ahead country-wide wind power prediction for Germany using gated recurrent unit deep learning. *Renew. Sustain. Energy Rev.* **2022**, *167*, 112700. [CrossRef]
22. Busch, J.; Kocheturov, A.; Tresp, V.; Seidl, T. NF-GNN: Network flow graph neural networks for malware detection and classification. In Proceedings of the 33rd International Conference on Scientific and Statistical Database Management, Tampa, FL, USA, 6–7 July 2021; pp. 121–132.
23. Bilgic, M.; Mihalkova, L.; Getoor, L. Active learning for networked data. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 79–86.
24. Moore, C.; Yan, X.; Zhu, Y.; Rouquier, J.B.; Lane, T. Active learning for node classification in assortative and disassortative networks. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 21–24 August 2011; pp. 841–849.
25. Cai, H.; Zheng, V.W.; Chang, K.C.C. Active learning for graph embedding. *arXiv* **2017**, arXiv:1705.05085.
26. Gao, L.; Yang, H.; Zhou, C.; Wu, J.; Pan, S.; Hu, Y. Active discriminative network representation learning. In Proceedings of the IJCAI International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018.
27. Liu, J.; Wang, Y.; Hooi, B.; Yang, R.; Xiao, X. LSCALE: Latent Space Clustering-Based Active Learning for Node Classification. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Grenoble, France, 19–23 September 2022; pp. 55–70.
28. Regol, F.; Pal, S.; Zhang, Y.; Coates, M. Active learning on attributed graphs via graph cognizant logistic regression and preemptive query generation. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 8041–8050.

29. Wu, Y.; Xu, Y.; Singh, A.; Yang, Y.; Dubrawski, A. Active learning for graph neural networks via node feature propagation. *arXiv* **2019**, arXiv:1910.07567.
30. Gilhuber, S.; Busch, J.; Rotthues, D.; Frey, C.M.; Seidl, T. DiffusAL: Coupling Active Learning with Graph Diffusion for Label-Efficient Node Classification. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Turin, Italy, 18 September 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 75–91.
31. Zhao, L.; Akoglu, L. Pairnorm: Tackling oversmoothing in gnns. *arXiv* **2019**, arXiv:1909.12223.
32. Sun, J.; Cheng, Z.; Zuberi, S.; Pérez, F.; Volkovs, M. Hgcf: Hyperbolic graph convolution networks for collaborative filtering. In Proceedings of the Web Conference 2021, Ljubljana Slovenia, 19–23 April 2021; pp. 593–601.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
34. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
35. Aggarwal, A.; Mittal, M.; Battineni, G. Generative adversarial network: An overview of theory and applications. *Int. J. Inf. Manag. Data Insights* **2021**, *1*, 100004. [CrossRef]
36. Yang, L.; Zhang, Z.; Song, Y.; Hong, S.; Xu, R.; Zhao, Y.; Zhang, W.; Cui, B.; Yang, M.H. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.* **2022**, *56*, 105. [CrossRef]
37. Li, P.; Chien, I.; Milenkovic, O. Optimizing generalized pagerank methods for seed-expansion community detection. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019.
38. Kloumann, I.M.; Ugander, J.; Kleinberg, J. Block models and personalized PageRank. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 33–38. [CrossRef]
39. Berberidis, D.; Giannakis, G.B. Node embedding with adaptive similarities for scalable learning over graphs. *IEEE Trans. Knowl. Data Eng.* **2019**, *33*, 637–650. [CrossRef]
40. Faerman, E.; Borutta, F.; Busch, J.; Schubert, M. Semi-supervised learning on graphs based on local label distributions. *arXiv* **2018**, arXiv:1802.05563.
41. Borutta, F.; Busch, J.; Faerman, E.; Klink, A.; Schubert, M. Structural graph representations based on multiscale local network topologies. In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, Thessaloniki, Greece, 14–17 October 2019; pp. 91–98.
42. Gasteiger, J.; Bojchevski, A.; Günnemann, S. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv* **2018**, arXiv:1810.05997.
43. Faerman, E.; Borutta, F.; Busch, J.; Schubert, M. Ada-LLD: Adaptive Node Similarity Using Multi-Scale Local Label Distributions. In Proceedings of the 2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), Melbourne, Australia, 14–17 December 2020; pp. 25–32.
44. Busch, J.; Pi, J.; Seidl, T. PushNet: Efficient and adaptive neural message passing. *arXiv* **2020**, arXiv:2003.02228.
45. Gasteiger, J.; Weißenberger, S.; Günnemann, S. Diffusion improves graph learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019.
46. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
47. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
48. Zeng, H.; Zhou, H.; Srivastava, A.; Kannan, R.; Prasanna, V. Graphsaint: Graph sampling based inductive learning method. *arXiv* **2019**, arXiv:1907.04931.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.