Front. Eng. Manag. 2025, 12(4): 736–753
https://doi.org/10.1007/s42524-025-4030-5

RESEARCH ARTICLE

Jun YIN, Rutao MA, Shilun GE

Predicting task bottlenecks in digital manufacturing enterprises based on spatio-temporal graph convolutional networks

Abstract Digital manufacturing enterprises require high operational agility due to the intricate and dynamically changing nature of their tasks. The implementation of accurate and timely predictions of task bottlenecks is therefore crucial to enhancing overall efficiency. Due to task complexities and dynamic business environments, bottleneck prediction is a challenging issue. This study introduces a novel approach that constructs a task network from extensive data accumulated within a digital enterprise to identify and depict the complex interrelations among tasks. Based on this method, we develop a Bottleneck Spatio-Temporal Graph Convolutional Network (BTGCN) model based on deep learning methods that considers spatial features of the task network and temporal data of task execution and integrates the strengths of GCN and GRU. We find that GCN effectively learns and represents the complex topology of task networks to capture spatial dependencies, while GRU adapts to the dynamic changes in task data, accurately capturing temporal dependencies. Informed by the theory of constraints, the study applies the proposed BTGCN model to the prediction of task throughput bottlenecks in digital enterprises. Experimental results demonstrate that while the model has certain limitations, it can accurately extract spatio-temporal correlations from system data, offering advantages in bottleneck prediction over other benchmark models.

Keywords task bottleneck, task network, digital manufacturing enterprise, spatio-temporal

Received Feb. 20, 2024; revised Oct. 29, 2024; accepted Nov. 15, 2024

Jun YIN, Rutao MA (✉), Shilun GE
College of Economics and Management, Jiangsu University of Science and Technology, Zhenjiang 212000, China
E-mail: mptgg@outlook.com

This research was supported by the National Natural Science Foundation of China (Grant Nos. 72432004 and 72372060)

1 Introduction

In the realm of digital manufacturing enterprises, enhancing overall business efficiency is crucial to operations when business processes undergo digital transformation. Achieving this objective necessitates the identification of key constraints that impede business efficiency, followed by a systematic improvement of these conditions to boost the performance of the entire business process. Within the manufacturing sector, these critical constraints are often referred to as "bottlenecks" (Huang et al., 2014). In the context of manufacturing, the foundational work of Goldratt and Cox (1984) defined bottleneck or contraint as the slowest operation or machine that determines the speed at which products can be produced and the maximum rate of output and throughput, wherein any time lost in production on the bottleneck is time lost for the entire business. As the development of Industry 4.0 and Industry 5.0 progresses, the concept of lean production, where identifying and eliminating bottlenecks is a key goal, has been further expanded and deepened (Xiang et al., 2024). Industry 4.0—through the introduction of emerging technologies such as the Internet of Things (IoTs), big data, and artificial intelligence (AI)—has realized the automation and intelligentization of production processes (Hajlaoui et al., 2024), thereby enhancing the flexibility and efficiency of manufacturing systems. Industry 5.0, on the other hand, emphasizes human-machine collaboration (Piccarozzi et al., 2024), placing greater emphasis on human leadership and personalized customization in foundational processes of intelligent automation.

While the advancement of these technologies facilitates the optimization of manufacturing systems, it also increases the complexity of identifying and addressing bottlenecks. In the context of Industry 4.0/5.0, lean production must therefore go beyond its traditional focus on waste reduction and efficiency improvement to consider how to leverage new technologies to identify,

predict, and eliminate bottlenecks (Chang and Jia, 2023) and manage throughput. Drawing on Goldratt and Cox's (1984) highly influential theory of constraint (TOC), we define a bottleneck as the task within a system that has the lowest throughput. Throughput refers to the task execution frequency per unit time, serving as a critical measure of a task's efficiency and speed.

The timely identification and accurate prediction of bottlenecks are crucial for optimizing business processes and supporting enterprise decision-making, yet bottleneck prediction poses a significant challenge for organizations. This is primarily due to the complex spatio-temporal characteristics of bottlenecks related to spatial dependency and temporal dependency. Spatial dependency refers to the tendency for nearby locations to possess similar characteristics and influence each other. In the context of enterprise operations, spatial dependency manifests in the throughput changes of tasks that are influenced by the topology of the task network. The dependency relationships between different system tasks define their spatial structure, where these structural connections lead to mutual influences among tasks.

Moreover, the multi-faceted interrelations among tasks are not limited to direct dependencies but also include various forms such as resource sharing, workflow overlap, and indirect impacts (Kiggundu, 1981). The interactive nature of these relationships implies that any change in a single task can create a cascading effect throughout the entire network, as noted in Fig. 1. In the context of production, this means that even if two tasks are not directly connected to each other in the process, they may still be mutually influential due to their sharing of resources or equipment. Consequently, bottlenecks might not be localized; an interruption or bottleneck in one part of the network can propagate and lead to more interruptions or inefficiencies in other parts of the network, resulting in a domino effect where the initial problem exacerbates as it spreads throughout the system. Understanding and mitigating these cascading effects is important for maintaining smooth and efficient manufacturing operations. Assessing the spatial structure and interactions between tasks can therefore help businesses more accurately identify and predict bottlenecks.

Temporal dependency refers to the relationship of events or data points over time, and how past events can impact future outcomes. Considering temporal dependency is essential to operational efficiency in relation to two key functions: bottleneck prediction and trend identification. Accurate prediction of bottlenecks is nevertheless challenging, as they are not static and can change over time based on production schedules, maintenance activities, and varying demands. Analyzing temporal data enables the identification of recurring trends and patterns, such as bottlenecks that recur at specific times or in response to particular operational conditions. This understanding allows enterprises to enact proactive measures to prevent future bottlenecks. As task throughput can also dynamically change over time, it exhibits clear periodic trends that can affect product planning, as seen in Fig. 2. That is, the throughput at the current moment may be influenced by the execution status of tasks at the previous moment or even earlier, exemplified by the actual dispatch tasks in material management, as shown in Fig. 3. This temporal dependency requires the use of models capable of capturing

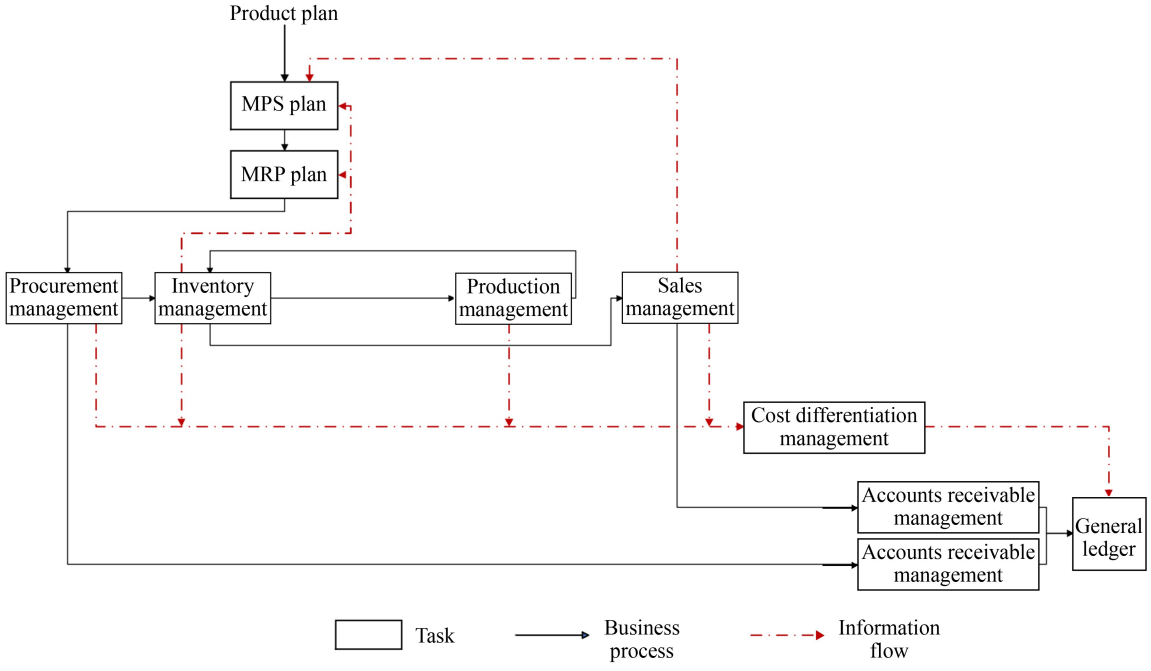

Fig. 1 Complex interrelations among tasks.

the dynamic characteristics of time series data to predict future throughput.

Extensive research conducted on bottleneck identification and prediction has identified a close correlation between the systemic bottleneck conditions and their historical observations. To effectively identify and quantify these potential bottleneck conditions, studies usually adopted decision analysis methods (Li, 2018) and simulation approaches (Li et al., 2011). Recent research on bottleneck prediction has primarily employed a time series analysis perspective, wherein predictions are made by studying historical data (Lai et al., 2018; Fang et al., 2020). These time series models have achieved high accuracy in bottleneck prediction, providing significant tools and methodologies for bottleneck forecasting.

Although bottleneck prediction methods based on time series data have demonstrated significant advantages in multiple application scenarios, they still exhibit limitations in providing comprehensive information for future scheduling and planning. Previous related research has primarily focused on accurately predicting potential bottleneck tasks within systems, yet they often failed to fully consider the complex spatial relationships and dynamic changes of tasks in business processes. In summary, these shortcomings are found in three main areas related to linear, temporal, and task relationships:

1) Bottleneck prediction models based on linear or simplistic assumptions, which do not address how bottlenecks often involve complex nonlinear relationships, can lead to a failure in capturing the true bottleneck conditions, thereby reducing the accuracy of predictions;

2) Research methods focused solely on the temporal aspect of bottlneck prediction may exhibit certain limitations in some scenarios, as bottlenecks often involve multi-factorial and multidimensional issues. For complex systems, it might be challenging to capture various

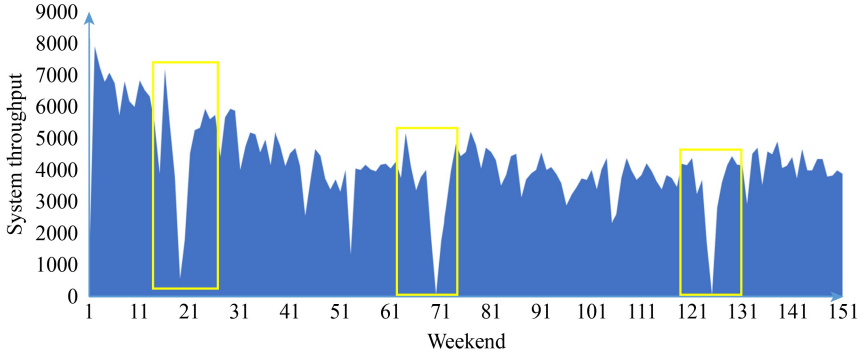

Fig. 2 Weekly variations in task throughput.

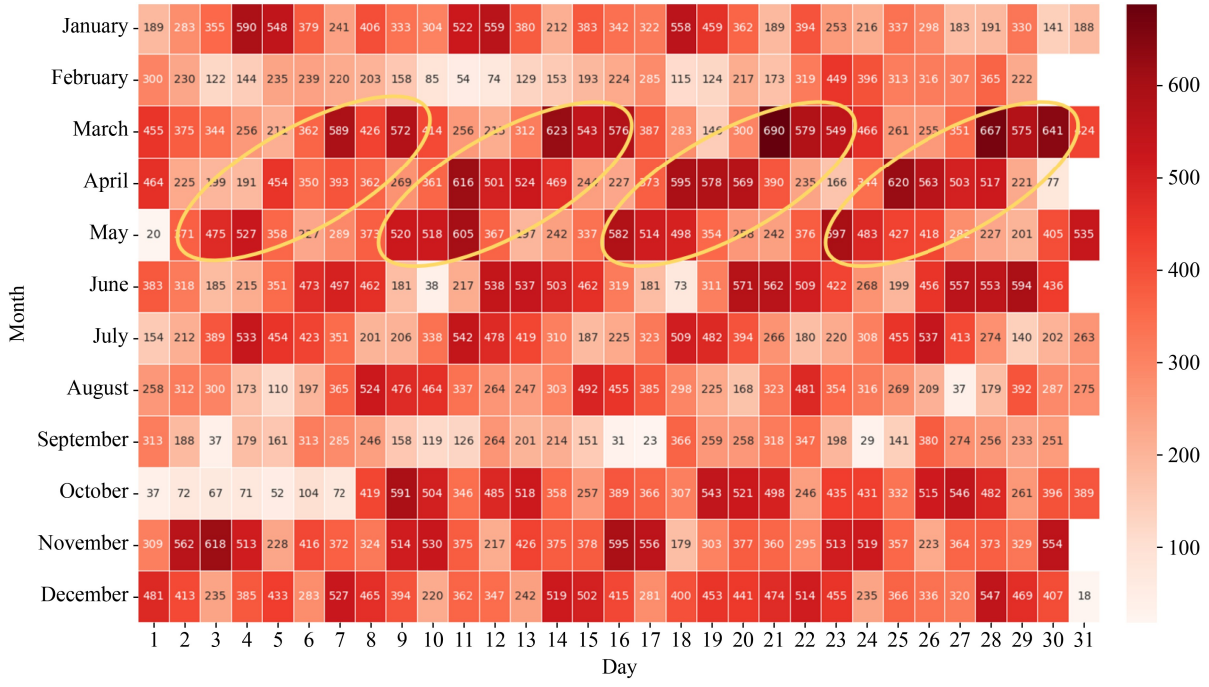

Fig. 3 Fluctuations in throughput of actual dispatch tasks in material management.

interactive effects and complex causal relationships using a temporal focus;

3) The critical role of interactions between related tasks in triggering bottlenecks suggests that understanding the interplay between different tasks is crucial for the formation and resolution of bottlenecks. To date, research on these spatial relationships and their interactions in bottleneck prediction has been limited.

To address these research gaps, this study introduces two key network approaches:

1) Considering the complex nonlinear relationships between tasks, we construct a task-oriented network to depict the spatial interactions among tasks. This network structure can reveal and quantify the interactions between tasks, capture spatial features, and reflect the complex connections between tasks, laying the foundation for subsequent bottleneck prediction;

2) From a spatio-temporal perspective, based on graph neural networks and utilizing time series data along with network spatial feature data, we introduce a Bottleneck Spatio-Temporal Graph Convolutional Network (BTGCN) to more accurately predict task bottleneck issues related to complex manufacturing enterprises in digital settings.

Informed by TOC, also known as bottleneck theory (Goldratt and Cox, 1984), this study quantitatively analyzes task throughput to identify critical bottleneck tasks. Our construction of the novel task-based network that incorporates spatial task characteristics and combines time series with spatial task data based on deep learning methods forms the foundation of the proposed BTGCN model, offering practical decision-making support for optimizing enterprise business processes.

2 Related work

2.1 Advances in bottleneck identification and prediction

In digital manufacturing enterprises, bottlenecks refer to tasks that impose the greatest limitations or constraints on overall system performance (Li, 2018). Research in the field of bottleneck identification has primarily utilized three approaches: decision analysis, simulation-based approaches, and data-driven methods. Decision analysis employs mathematical models and statistical techniques to aid enterprise managers in gaining a deeper understanding of bottleneck issues and addressing their impact on manufacturing processes. Yet these techniques often rely on assumptions that may not be applicable in complex and dynamic manufacturing environments. Simulation-based methods, on the other hand, create virtual models of systems to simulate their operation and assist in identifying potential bottlenecks.

While these methods can offer valuable insights, they are often time consuming and may not accurately capture real-time change. Compared to other methods, data-driven analysis—with its minimal reliance on assumptions and high adaptability to complex data patterns—supports more flexible and scalable applications (Lai et al., 2021). It thus shows greater potential in handling real-time data and predicting future trends while minimizing reliance on assumptions, making it highly adaptable to complex data patterns. Nevertheless, studies have also noted that the location of bottlenecks in business processes often shifts among different tasks, exhibiting dynamic variability (Subramaniyan et al., 2020). The existing methods for bottleneck identification have primarily focused on detecting real-time bottlenecks or tracing historical bottleneck information (Cao et al., 2012), which limits their effectiveness in long-term operational planning and the achievement of efficient business processes.

As digital manufacturing enterprises evolve, the integration of management information systems and automated data collection technologies has led to the accumulation of vast amounts of data (Wu et al., 2017). These data provide comprehensive insights into business processes, enabling researchers and businesses to analyze and identify key factors affecting efficiency, thereby more accurately pinpointing potential bottleneck areas. While these insights contribute to the field, the identification of bottleneck conditions at specific locations is often related to past observations. To capture this temporal dependency, Autoregressive Integrated Moving Average (ARIMA) time series forecasting has been employed to predict bottlenecks (Li et al., 2011), along with the use of active periods as an indicator of such bottlenecks (Subramaniyan et al., 2018a). Moreover, several factors are found to directly impact the dynamics of production lines, such as product mix, release order, and operator shifts. To address this issue, an adaptive network-based fuzzy inference system has been introduced to predict future bottlenecks based on production inputs (Cao et al., 2012).

To further capture input-output relationships and temporal trends of the system, research has explored models based on the deep learning models used recurrent neural networks (RNNs). For instance, Bidirectional Long Short-Term Memory (BiLSTM), a type of RNN model that processes sequential data in both forward and backward directions, which was developed to predict system congestion and starvation based on high-dimensional multivariate time series data (Lai et al., 2018). Another RNN method, the Gated Recurrent Unit (GRU), which enhances the speed performance of LSTM networks, which was proposed for shifting lateness bottleneck prediction (Fang et al., 2020). Despite such advancements, these methods primarily address temporal dependency without fully incorporating spatial relationships. Without considering spatial factors, models may fail to capture critical interactions among production units, resulting in less effective interventions and planning.

This approach thus undermines the accuracy and effectiveness of bottleneck prediction models, which can lead to incomplete analyses and suboptimal predictions.

Specifically, the prevailing approach in existing research has been to predict bottleneck indicators, such as active periods, congestion, and starvation, for a future period and then apply existing bottleneck identification algorithms to determine future bottlenecks (Yu and Matta, 2016). The current bottleneck prediction methods have primarily focused on capturing the temporal evolution trends of systems, usually by independently predicting each task's bottleneck indicators, where the potential impact of spatial information on bottleneck conditions is often overlooked. Yet the propagation of bottleneck impacts caused by complex topologies between mission systems also plays an important role in bottleneck prediction. Hence, there is a need to further explore how to effectively integrate information from both temporal and spatial dimensions to more accurately predict and manage bottleneck occurrences in various systems, thereby improving their overall efficiency and performance. We therefore identify and elucidate the complex interrelationships among tasks by constructing task networks from the large amount of data accumulated by a digital enterprise over a period of nearly three years. This allows us to capture the spatial and temporal feature information of tasks more accurately, where our use of deep learning techniques helps achieve accurate bottleneck prediction.

2.2 Research on graph neural networks for capturing temporal and spatial features

Graphs consisting of edges and vertices represent a highly powerful form of data representation, allowing the abstract depiction of systems comprising interacting objects and their intricate spatial relationships (Hu et al., 2020). Numerous research domains rely on graphs to model the structures of various entities and their associations, where their applications include social networks (Rezvanian and Meybodi, 2016; Li et al., 2023b), web and knowledge graphs (Fionda et al., 2016; Ye et al., 2022), and transportation networks (Sun et al., 2021). Notably, the field of machine learning for processing graph-structured data has witnessed significant breakthroughs. For example, Traditional Convolutional Neural Networks (CNNs) excel at handling Euclidean space data (Zhao et al., 2019), and are particularly skilled in tasks with a regular grid structure, such as image classification (Krizhevsky et al., 2017) and machine translation. Yet CNNs have numerous limitations when dealing with non-Euclidean space data, such as graph-structured data. For instance, CNNs rely on regular grid structures for convolution operations, but graph data are typically irregular, where its complex connections between nodes and edges may exhibit no fixed patterns (Monti et al., 2017), which makes traditional convolution operations difficult to apply directly to graph data. Moreover, CNNs lack an understanding of graph topology, and thus fail to effectively capture the complicated relationships between nodes and the global features of the graph. CNNs are also susceptible to changes in graph structure and have poor adaptability to dynamic changes and topology updates, limiting their application in tasks that require the handling of dynamic graph data.

In contrast, Graph Neural Networks (GNNs) have attracted attention for their superior ability to capture graph-structured data. GNNs can effectively understand and represent the complex relationships between nodes and their features, while also processing information synchronously within the nodes' neighborhoods (Peng et al., 2022). This advanced capability has made GNN an effective and popular tool in many fields, especially in dealing with tasks that require the analysis and processing of complex relationships between nodes (Wu et al., 2020). Furthermore, the characteristics of GNN offer a new perspective and approach for machine learning in handling graph-structured data, driving deeper and broader research across various domains.

In traditional GNN models, such as graph convolutional networks (GCNs) (Li et al., 2023a) and GraphSAGE (Zhang et al., 2022), an equitable feature aggregation method for adjacent nodes is adopted to compute the representation of the central node. The advantage of this method is its equal integration of node information, providing a comprehensive feature view for the central node. This balanced approach ensures fairness and consistency in capturing spatial relationships between nodes within the model, thereby effectively avoiding over-reliance on or bias toward specific nodes. Furthermore, this equitable feature aggregation strategy simplifies the learning process of the model and reduces its complexity, making GNN more efficient in processing large-scale graph data. In many application scenarios, such as social network analysis and traffic networks, this method has been proven to effectively capture and represent the global structural features of the graph (Buffelli and Vandin, 2022), thereby providing a robust and reliable performance foundation for related tasks. Nevertheless, traditional GNN models primarily focus on the spatial relationships between nodes and their neighbors, and are unable to effectively capture the dynamic temporal features of nodes. This limitation becomes significant when dealing with time-related tasks, such as predicting enterprise states or behavioral changes. As traditional GNNs face challenges in handling time series data, they cannot fully leverage temporal information to improve prediction accuracy.

The proposed BTGCN task network also exhibits certain similarities to traffic or social networks, with each task (node) having unique features, interacting spatially with adjacent tasks while being related temporally to its historical observations. We therefore employ BTGCN for

more precise task bottleneck predictions, based on the spatial features of the task network and the temporal data of task execution. By applying advanced neural network technology within the complex environment of the manufacturing domain and accurately capturing the correlations of temporal and spatial information, the BTGCN model has important advantages in addressing these prediction challenges.

3 Methodology

3.1 Problem statement

In the era of digital manufacturing, the efficiency and responsiveness of task execution have become central to competitiveness in the industry(Ahuett-Garza and Kurfess, 2018). With the variability of market demands, the efficient mitigation of task bottlenecks has emerged as a critical factor in system performance. The development of digital manufacturing enterprises has led to a huge increase in the amount of system data (Chryssolouris et al., 2009), enabling a comprehensive depiction of the enterprise task process while providing a data foundation for identifying and interpreting complex task associations. Traditional bottleneck prediction methods based on time series data, which have shown limitations in addressing complex business processes and fluctuating market demands—often assume the system is static. This approach overlooks how in actual operations, system states are dynamically changing; not only do bottlenecks drift over time, but bottleneck congestion locations also shift due to changes in task spatial associations (Biller et al., 2009). A more flexible and dynamic approach is therefore invaluable for capturing the real-time state of the system and more accurately predicting potential bottlenecks. This study examines how to quantify the spatial associations between tasks and fuse them with temporal features applied to bottleneck identification and prediction. To achieve this objective, it is necessary not only to conduct in-depth analyses of time series data that capture the temporal dynamics of task execution, but also to consider the implications of spatial features between tasks.

3.2 Problem definition

This research is centered on bottleneck prediction, with the aim of leveraging historical task operation data from the system to predict throughput bottlenecks within a designated time window. Our methodology entails the construction of spatial relationships among tasks to establish an adjacency matrix, while concurrently employing task execution frequency as a metric to quantify throughput, thus facilitating the creation of a feature matrix. Through this level of meticulous analysis and computation, it is possible to effectively predict and manage bottleneck issues. The problem definition used for our analytical process consists of two parts: Task Network G and Feature Matrix $X^{N\times P}$.

Definition 1: Task Network G. We define a task network as an unweighted graph $G=(V,E)$ to characterize the topological structure between tasks. Each task herein is considered as a node within the graph, with the set $V=\{v1,v2,...,vN\}$ representing all task nodes, where N denotes the total number of nodes. E represents the set of edges between nodes, and the adjacency matrix $A\in R^{N\times N}$ describes the connections between tasks, consisting of 0s and 1s to reflect the direct contact between nodes.

Definition 2: Feature Matrix $X^{N\times P}$. Within this framework, the task throughput information on the task network is regarded as attribute features of the network nodes, represented by the feature matrix $X\in R^{N\times P}$, where N represents the number of nodes, and P represents the number of attribute features for each task node (the length of historical time series). $X_t\in R^{N\times m}$ is used to denote the execution frequency of each task node at m time points.

Our two-part definition enables us to precisely capture and analyze the dynamic behaviors of various task nodes across different time series, thereby optimizing the overall performance of the network. This approach allows us to effectively predict and adjust task nodes to accommodate future workload variations, thereby enhancing the overall throughput and efficiency of system tasks.

The spatio-temporal bottleneck prediction problem can thus be viewed as the process of learning a mapping function f, given a task network G and its corresponding feature matrix X, with the aim of predicting the execution frequency information for the next T time points, as shown in Eq. (1):

$$X_{t+1},\ X_{t+2},\ \ldots,\ X_{t+T}=f\left(G;(X_{t-p},\ \ldots,\ X_{t-1},\ X_t)\right), \quad (1)$$

wherein T defines the number of future time points for which the model makes predictions, such as how many time points into the future the prediction extends from the current moment, and p defines the time window of the time series used for prediction, such as how many past time points' data are considered in making the prediction.

3.3 Methods

In applying BTGCN to task bottleneck prediction, the model integrates GCN and GRU, where the GCN component analyzes the complex topology of the task network to capture spatial dependencies and extract spatial features, and the GRU component handles time series data with spatial features. This method captures temporal dynamics through inter-unit information transfer mechanisms and the extraction of temporal features. Specifically, the process involves constructing a task network to identify interrelated tasks, obtaining the necessary spatial adjacency

matrix data, and then using GCN to capture the spatial features among these tasks. Next, based on the task's time series data, we use GRU to extract temporal features. Finally, the fully connected layer of the BTGCN model performs comprehensive analysis to generate prediction results. Figure 4 illustrates this four-step prediction process more intuitively.

3.3.1 Task network

In our proposed task network (which demonstrates similar characteristics to traffic or social networks), each task is considered a unique node with specific task features. These tasks interact with adjacent ones on a spatial level and are associated with their historical execution records on a temporal dimension. To construct this task network, we collect and pre-process the enterprise resource planning (ERP) system data from a large ship-building company over a period of two years and 10 months. Identifying tasks processed by the same operator enables our identification of interrelations among the various tasks. Based on these associations, we construct a task network matrix, effectively depicting the spatial connections between tasks, where the construction process in three steps is illustrated in Fig. 5.

3.3.2 Graph convolutional network (spatial dependency modeling)

In digital manufacturing enterprises, the complex interrelations and mutual influences among tasks determine the efficiency and timeliness of operations. To accurately predict potential bottlenecks, it is crucial to identify and model these complex interactions. While traditional CNNs excel at extracting local spatial features, they are primarily suited for regular Euclidean data structures, such as images and regularly arranged grids. For task networks represented as irregular graph structures, this limitation makes it challenging for CNNs to accurately reveal complex topological connections, thus failing to fully capture the spatial dependencies between tasks.

In recent years, the emergence of GCN has provided new insights for addressing these issues. GCN has demonstrated outstanding performance on non-Euclidean data, where it can adapt to various complex graph structures (Bronstein et al., 2017). By defining filters in the Fourier domain and applying them to the nodes of a graph, GCN considers the information of each node and its surrounding neighborhood, thereby effectively capturing spatial characteristics. This approach has shown broad application potential in fields such as node classification (Parisot et al., 2018), relationship prediction (Zhang et al., 2019), and graph-structured data (Vashishth et al., 2019). Specifically, GCN inputs the adjacency matrix A and feature matrix X, utilizing a multi-layer convolutional structure to delve into and learn the spatial dependencies within the graph, thus offering a powerful analytical tool for tasks like document classification, social network analysis, and image recognition. With the provision of adjacency matrix A and feature matrix X, the GCN model establishes filters in the Fourier domain to perform feature analysis of graph nodes. These filters focus on

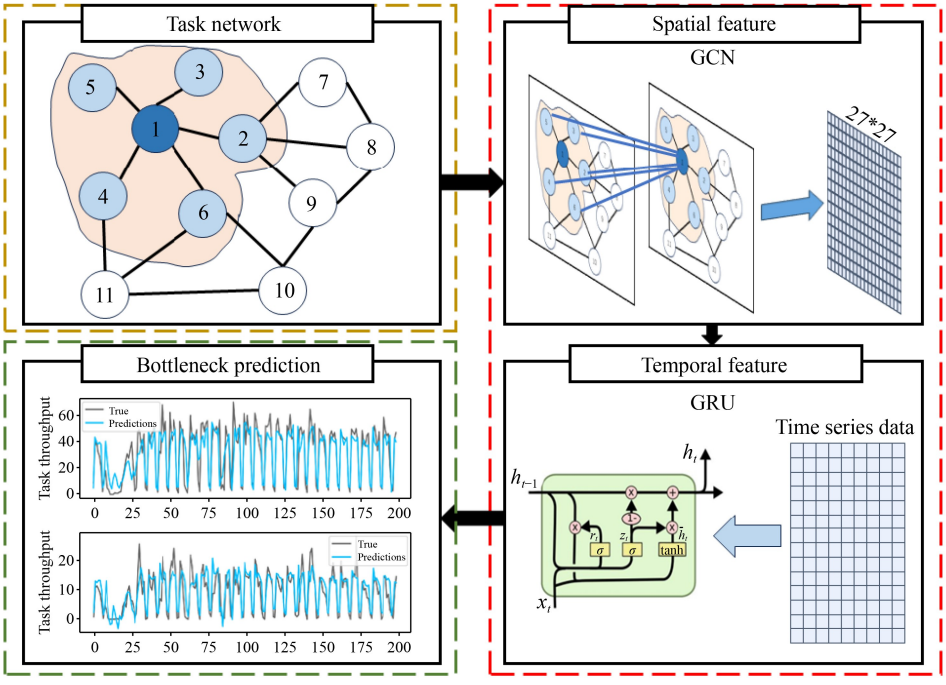

Fig. 4 Bottleneck prediction methodology flowchart.

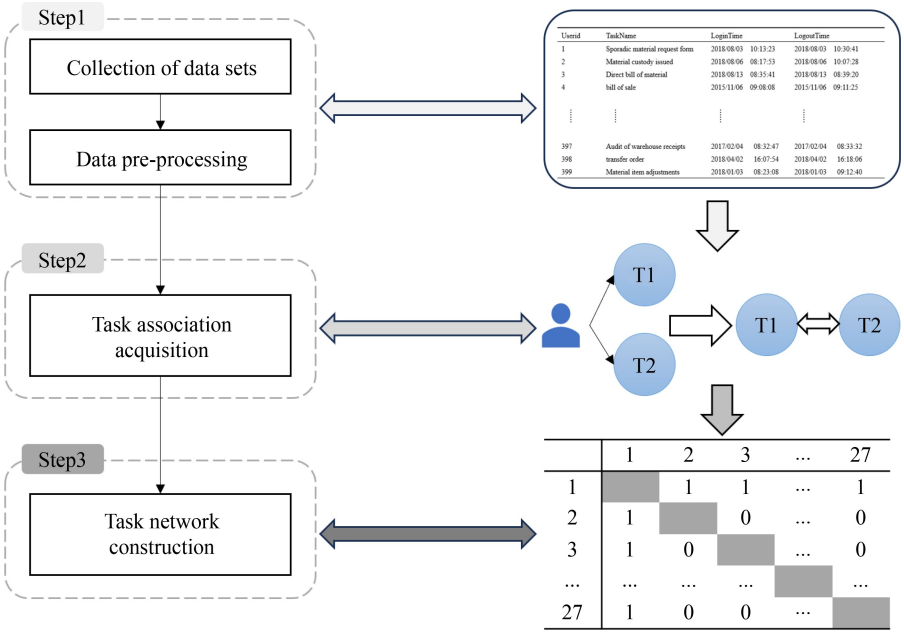

Fig. 5 Task network construction methodology.

each node in the graph and its directly connected neighborhood, meticulously extracting spatial connectivity characteristics. By stacking several convolutional layers, GCN can progressively deepen the understanding and representation of the graph structure, effectively capturing the complex spatial relationships between nodes. The basic hierarchical relationship can be represented as

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

wherein $H^{(l+1)}$ is the node feature matrix at layer $l+1$ (for $l=0$, $H^{(0)}=X$, the input feature matrix), σ represents the sigmoid function in the nonlinear model, $\tilde{A}=A+I_N$ is the adjacency matrix A plus the identity matrix I_N to account for self-connections, $\tilde{D}$ is the degree matrix of $\tilde{A}$, where $\tilde{D}_u=\sum_j\tilde{A}_{ij}$, and $W^{(l)}$ is the weight matrix at layer l. In this study, we employ a two-layer GCN model to capture spatial dependencies, with its formulation given as

$$f(X,A) = \sigma(\hat{A}\mathrm{PReLU}(\hat{A}XW_0)W_1), \tag{3}$$

wherein $f(X,A)$ takes the graph's feature matrix X and adjacency matrix A as inputs, representing the output of the entire GCN model. $\hat{A}$ denotes the pre-processed adjacency matrix, PReLU is used as the nonlinear activation function, and $PReLU(\hat{A}XW_0)$ signifies the first layer of graph convolution operation, where W_0 is the weight matrix from the input layer to the hidden layer, and W_1 represents the weight matrix from the hidden layer to the output layer.

In this basic GCN model, each layer normalizes its input through $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$, then multiplies it with the weight matrix and applies a nonlinear activation function. Hence, each layer is capable of effectively capturing and integrating the local neighborhood information of nodes, progressively constructing a deep understanding of the entire graph structure. By stacking multiple layers of such structure, GCN can capture the complex spatial dependencies between nodes.

3.3.3 Gated recurrent unit (temporal dependency modeling)

In the complex environment of enterprises, tasks exhibit significant temporal correlations and interdependencies. To effectively model and predict such temporal characteristics, we employ GRU, an efficient neural unit designed specifically for capturing dependencies in time series data (Shen et al., 2018). The variation in bottleneck locations is a result of combined effects of the task network's spatial dynamics, its historical time series data, and enterprise planning. In this context, RNN provides a widely applied methodological framework for processing sequence data. Nevertheless, RNNs are limited by the vanishing and exploding gradient problem when handling long-term dependencies (Arjovsky et al., 2016). Serving as an improvement, LSTM and GRU introduce gating mechanisms, significantly enhancing the model's ability to capture temporal dependencies (Tallec and Ollivier, 2018). Although LSTM and GRU share the core concept of gating mechanisms in their structure, GRU has been shown to be more efficient in many contexts due to having fewer parameters and a simpler architecture (Zhang et al., 2018; Kisvari et al., 2021). Employing a GRU model allows us to capture the current moment's bottleneck throughput while retaining the trend of historical bottleneck throughput variations, thus providing the

capability to capture temporal correlations. Given the predictive task requires integrating the entire network structure information, it is imperative to consider the embedding information of all nodes in the graph. To accurately predict and understand the global behavior of executing tasks, it is therefore necessary to integrate the embedding information of various nodes, wherein construction of a graph-level representation reflects the entire task network's characteristics, as follows:

$$H_{\text{graph}} = \text{AGGREGATE}(H^{(1)}, H^{(2)}, \ldots, H^{(N)}), \quad (4)$$

wherein H_{graph} represents the graph-level features, $H(i)$ denotes the embedding of the ith node, and AGGREGATE is the aggregation function, employed to implement the aggregation operation of the graph neural network.

The selection of a global aggregation function should be optimized based on the requirements of the specific application scenario. For instance, global sum pooling may offer superior performance when the focus is on the scale of nodes within the graph. We employ global max pooling to construct the graph-level feature H_{graph}. This method effectively captures key spatial attributes by filtering the most significant features across each embedding dimension. The constructed H_{graph} is subsequently used as input to GRU, facilitating the model's capability to capture and integrate the dynamics of time series data. Each learning unit of the GRU cell uses the following Equations:

$$z_t = \sigma(W_z \cdot [h_{t-1}, H_{\text{graph}}]), \quad (5)$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, H_{\text{graph}}]), \quad (6)$$

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, H_{\text{graph}}]), \quad (7)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t, \quad (8)$$

wherein z_t and r_t are the update gate and reset gate vectors at the current time step, respectively, determining how much information from the previous moment needs to be retained or ignored. W_z, W_r, and W_h are the weight matrices for the update gate, reset gate, and candidate hidden state, respectively, and h_{t-1} represents the hidden state from the previous time step.

3.3.4 Bottleneck spatio-temporal graph convolutional network

To comprehensively exploit the spatial layout and temporal dynamics of executing tasks in digital manufacturing enterprises, the proposed BTGCN model integrates GCN and GRU. The core innovation of BTGCN lies in its capability to simultaneously parse the spatio-temporal bottleneck characteristics of system tasks. By merging the spatial feature extraction ability of GCN with the time series processing advantage of GRU, the model achieves a deep understanding and accurate prediction of complex business processes. Furthermore, its adaptability and sensitivity enhance our ability to assess complex spatial dependencies and temporal evolution trends by introducing multi-scale feature aggregation and dynamic weight adjustment strategies. These characteristics enable BTGCN to effectively improve the accuracy and stability of predictions when dealing with high-dimensional and variable system task data.

In its architectural design, BTGCN initially utilizes GCN as a spatial feature extractor, optimizing its ability to analyze the nodes of tasks and their connectivity, thereby strengthening the model's recognition of key nodes and bottleneck areas. By capturing the complex relationships among nodes in space with GCN, the model effectively extracts deep features about the task structure. Subsequently, through the GRU layer acting as a temporal feature extractor, the model can capture long-term dependencies within the time series. Introduction of the GRU layer not only increases the model's memory capacity for historical data but also allows for dynamic adjustment of focus on historical information, thus improving the accuracy and robustness of predictions. Finally, by employing Multi-Layer Perceptron (MLP) as the prediction layer, spatial and temporal features are integrated to output the final bottleneck prediction results. To further enhance performance, BTGCN incorporates regularization and optimization strategies to ensure the model possesses good generalization ability and computational efficiency across diverse system environments. By using these strategies, BTGCN effectively adapts to various enterprise environments when handling high-dimensional and variable system task data, improving the accuracy and stability of predictions.

3.3.5 Loss function

The loss function employed by the BTGCN model is designed to ensure the accuracy of its predictions while preventing overfitting, comprising two parts. The first part is the sum of squared errors, represented as $\sum_{t=1}^{T}(Y_t - \tilde{Y}_t)^2$, which calculates the mean squared error between the model's predicted bottleneck feature values $\tilde{Y}_t$ at each time step t and the actual observed values Y_t, intuitively reflecting the model's predictive performance across the entire time series. The second part is the L2 regularization term, $\lambda \sum_l \|W^{(l)}\|_2^2$, which constrains the model's complexity by penalizing all weights $W^{(l)}$, where λ is the hyperparameter for regularization strength. This combined loss function, integrating error minimization and regularization, not only facilitates the model's learning on historical data, but also enhances the model's generalization capability by controlling the growth of weights,

thus improving its performance on unseen data. We use the following equation to calculate the data:

$$\mathcal{L}=\sum_{t=1}^{T}\left(Y_t-\tilde{Y}_t\right)^2+\lambda\sum_{l}\|W^{(l)}\|_2^2. \tag{9}$$

3.3.6 Bottleneck identification

In digital manufacturing enterprises, task execution frequency is commonly used to measure throughput, where the system's processing capacity is reflected by calculating the number of times a specific task is completed within a unit of time (Subramaniyan et al., 2018b). Specifically, throughput T_i is defined as the execution frequency of the ith task within a given time frame, with the total throughput of the system being the sum of all tasks' throughputs. Within this framework, bottleneck tasks can be identified as those with the lowest throughput, as they restrict the flow and efficiency of the entire system. In other words, bottleneck tasks (by definition) are completed more slowly relative to others, leading to production backlogs and decreased efficiency. By identifying and optimizing these bottleneck tasks, the performance of the whole system can be significantly improved, helping to uncover and address key issues limiting efficiency. This makes the measurement of throughput using task execution frequency and identification of bottleneck tasks an effective optimization strategy. The process of identifying bottlenecks in this study involves three steps:

1) Calculate the throughput of each task. For each task i in the system, we first calculate its throughput T_i, the task execution frequency per unit time:

$$T_i=\frac{\text{Execution frequency}}{\text{Time unit}}. \tag{10}$$

2) Calculate the total throughput. Subsequently, by calculating the sum of the throughput of all tasks, the total system throughput T_{system} is obtained, reflecting the overall processing capacity of the system:

$$T_{\text{system}}=\sum_{i=1}^{n}T_i. \tag{11}$$

3) Identify the bottleneck task. Finally, the task with the lowest throughput is identified as the bottleneck task (R. Thorne, 2006), primarily because in business processes or any type of flow, the output speed of the entire system is constrained by its slowest part. This concept is derived from TOC/bottleneck theory (Goldratt and Cox, 1984), where "bottleneck" refers to the narrow section that limits the overall performance of the system:

$$BT=\arg\min_{t}\{T_t\}. \tag{12}$$

In summary, understanding and identifying task bottlenecks are crucial for improving efficiency and performance. By pinpointing the tasks that limit overall throughput, resources and optimization efforts can be focused on enhancing these specific tasks, thereby improving the performance of the entire system.

4 Experiments

4.1 Data description

We evaluate the predictive performance of BTGCN model based on a data set from the ERP system of a large ship-building company, during a period spanning two years and 10 months. The data set comprises 727,665 records, covering 202 different management tasks, with a sample of the data shown in Table 1. Specifically, the analysis focuses on 27 key management tasks to explore their impact and characteristics within the business processes. The experimental data integrates both spatial relationships and time series dimensions: on one hand, a 27×27 adjacency matrix meticulously outlines the spatial dependency relationships between tasks, with matrix elements quantitatively reflecting the interconnectivity between tasks; on the other hand, the feature matrix records the execution frequency of each task, summarizing the execution frequencies over 12-h and 24-h cycles to

Table 1 Partial content of user's original operation data

Userid	TaskName	LoginTime	LogoutTime
1	Sporadic material request form	2018/08/03 10:13:23	2018/08/03 10:30:41
2	Material custody issued	2018/08/06 08:17:53	2018/08/06 10:07:28
3	Direct bill of material	2018/08/13 08:35:41	2018/08/13 08:39:20
4	Bill of sale	2015/11/06 09:08:08	2015/11/06 09:11:25
⋮	⋮	⋮	⋮
397	Audit of warehouse receipts	2017/02/04 08:32:47	2017/02/04 08:33:32
398	transfer order	2018/04/02 16:07:54	2018/04/02 16:18:06
399	Material item adjustments	2018/01/03 08:23:08	2018/01/03 09:12:40

reflect the periodicity and temporal patterns of tasks.

4.2 Data pre-processing

We initially conduct a data quality check, which includes data integrity verification, handling of missing values, and identification and processing of outliers. Given the extensive time span of the data, 12-h and 24-h intervals are chosen as time windows to better reflect the periodicity and trends in task execution. Using Python, the execution frequency of each task within each time window is then calculated from the raw data. Considering the execution frequency differences caused by the inherent characteristics of management tasks in manufacturing enterprises, task filtering is performed. Tasks with extremely low execution frequencies due to their intrinsic features—those with zero execution frequency in more than 50% of the time windows—are identified and excluded. Such tasks, due to their inherent properties, do not require frequent operation within the system and are unlikely to become bottleneck tasks. They contribute minimally to model learning and future prediction, and may introduce noise. After filtering, 27 tasks of high data quality and representativeness are selected for subsequent feature extraction and model training. Moreover, to comprehensively capture the structural characteristics between tasks, we construct a task network or graph $G = (V, E)$, where V and E represent nodes (management tasks) and edges (dependencies between tasks), respectively. Through this graphical representation, not only can the time series features of each task be extracted, but the spatial dependencies between tasks can also be obtained.

Following data filtering and graphical representation, we further normalize the selected task data to ensure effective model training and predictive performance. Specifically, we adopt the most commonly used min-max normalization (feature scaling) method, based on the following equation:

$$X_i' = \frac{X_i - \min(X)}{\max(X) - \min(X)}, \tag{13}$$

wherein X_i represents a specific value in the original data, $\min(X)$ and $\max(X)$ respectively denote the minimum and maximum values within the data set, and X_i' is the new value after normalization. Through this method, all task execution frequency data are transformed into a range from 0 to 1, enhancing the comparability between different tasks while also supporting stable training and rapid convergence of deep learning models.

4.3 Hyperparameter selection

4.3.1 Hyperparameters

Hyperparameters for the BTGCN model include: learning rate, batch size, training epochs, number of hidden units, and weight optimization, as shown in Table 2.

The number of hidden units has a crucial impact on the performance of the BTGCN model, as it determines the amount of information and complexity the model can capture. Different amounts of hidden units will directly affect the prediction accuracy. To identify the optimal number of hidden units, we conduct a series of experiments, systematically varying the number of hidden units. By comparing the prediction outcomes under different configurations, we can assess the impact of various numbers of hidden units on model performance and thereby determine the optimal number of hidden units that could provide the highest prediction accuracy, as shown in Fig. 6.

The left section of Fig. 6 demonstrates the performance variation of the BTGCN model with different hidden unit configurations based on 24-h cycle data, while the right section corresponds to the situation with 12-h cycle data. From these figures, we observe that when the cycle is set to 24 h, a hidden unit configuration of 200 enables the model to achieve optimal performance; correspondingly, for the 12-h cycle data, the model exhibits the best performance when the number of hidden units is set to 128. Based on these figures, we select the optimal number of hidden units corresponding to the different characteristics of periodic data, ensuring the model

Table 2 Hyperparameter tuning process

Hyperparameters	Range
Learning Rate	{0.1,0.01.**0.001**}
Batch Size	{16,**32**,64}
Epoch	{3000,4000,**5000**}
Number of Hidden Units	{64,100,**128**,**200**}
Weight Optimization	{SGD, RMSprop, **Adam**, Adagrad}

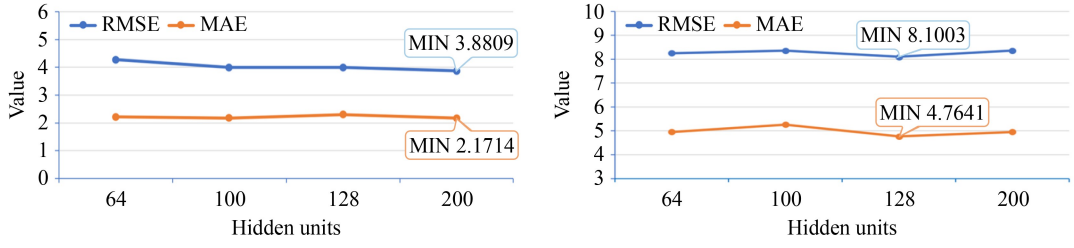

Fig. 6 Performance variation of BTGCN with different hidden units.

maximizes its predictive efficacy within the respective time ranges.

4.3.2 Training

For the configuration of the BTGCN model's input layer, 80% of the data are utilized as the training set during the model training phase to facilitate the model's learning and parameter optimization. The remaining 20% of the data are reserved as the test set, which is used to evaluate the model's generalization ability and predictive performance after training completion. The BTGCN model employs the Adam optimizer for model training. The Adam optimizer algorithm combines momentum and adaptive learning rate techniques, adjusting the update strategy for each parameter through the calculation of first-order moment estimates and second-order moment estimates of the gradients.

4.4 Comparison methods

This study compares the proposed BTGCN model with the following six commonly used bottleneck prediction methods:

1) History Average model (HA). The HA model can be used to predict bottleneck situations based on past performance. In bottleneck prediction, the HA model simply predicts the likelihood of a bottleneck at a future moment by calculating the average frequency of past bottleneck occurrences;

2) Autoregressive Integrated Moving Average (ARIMA). The ARIMA model is capable of modeling and predicting bottleneck situations in time series data, which is especially suitable for cases where bottlenecks have clear time series characteristics and periodicity;

3) Support Vector Regression (SVR). SVR constructs a hyperplane or sets of hyperplanes for predictions, with the goal of maximizing the margin between the actual and predicted outputs within a predetermined tolerance range;

4) Graph Convolutional Network (GCN). GCN learns representations of nodes by applying convolutional operations on the nodes of a graph, effectively capturing spatial relationships between nodes;

5) Gated Recurrent Unit (GRU). GRU introduces the concepts of update and reset gates to regulate the flow of information, allowing the network to capture long-distance dependencies in time series data; and

6) Bidirectional Long Short-Term Memory (BiLSTM). BiLSTM is a variant of the Long Short-Term Memory (LSTM) network. The BiLSTM model effectively predicts potential bottlenecks in the system by analyzing historical and future task execution patterns, thereby providing support for production scheduling, resource allocation, and other decisions.

4.5 Experimental results

We utilize three metrics to evaluate the predictive performance of the BTGCN model: MAE, RMSE, and R^2, calculated as follows:

1) Mean Absolute Error (MAE):

$$\mathrm{MAE}=\frac{1}{n_t\cdot n_s}\sum_{j=1}^{n_t}\sum_{i=1}^{n_s}\left|Y_{ij}-\widetilde{Y}_{tj}\right|. \tag{14}$$

2) Root Mean Squared Error (RMSE):

$$\mathrm{RMSE}=\sqrt{\frac{1}{n_t\cdot n_t}\sum_{j=1}^{n_s}\sum_{i=1}^{n_t}\left(Y_{ij}-\widetilde{Y}_{tj}\right)^2}. \tag{15}$$

3) Coefficient of Determination (R^2):

$$R^2=1-\frac{\sum_{j=1}^{n_s}\sum_{i=1}^{n_t}(Y_{ij}-\widetilde{Y}_{ij})^2}{\sum_{j=1}^{n_s}\sum_{i=1}^{n_t}(Y_{ij}-\overline{Y}_j)^2}, \tag{16}$$

wherein y_{ji} and $\tilde{y}_{ji}$ respectively represent the actual bottleneck information and predicted value for the ith task at the jth time sample, n_t is the number of time samples, n_s is the number of tasks, Y and $\tilde{Y}$, respectively, denote the sets of y_{ij} and $\tilde{y}_{ij}$, while $\overline{Y}$ is the average value of Y.

Table 3 presents a comparison of the performance of the BTGCN model and other baseline methods on the ERP system data set for 12-h and 24-h forecasting tasks. An asterisk (*) indicates a value so small as to be negligible, signifying poor predictive performance of the model. It can be observed that the BTGCN model achieves the best predictive performance under almost all evaluation metrics for all prediction ranges, demonstrating its effectiveness in spatio-temporal bottleneck prediction tasks in three key areas of overall predictive accuracy, spatial predictive capacity, and temporal predictive capacity.

The BTGCN model exhibits three primary advantages:

1) Higher overall predictive accuracy. Compared to traditional methods, the deep learning-based BTGCN model, GRU, and BiLSTM show significant advantages in predictive accuracy. When comparing baseline models such as HA, ARIMA, and SVR, neural network methods that are centered around time series feature modeling, like BTGCN, GRU, and BiLSTM, typically offer higher accuracy. For instance, in the 24-h throughput prediction task, the RMSE of BiLSTM, GRU, and BTGCN models decreased by approximately 4.61%, 2.65%, and 37.49%, respectively, compared to the HA model, while the MAE of the BTGCN model decreased by about 30.46% compared to the HA model. Compared to the ARIMA model, the MAE of GRU, BiLSTM, and BTGCN decreased by about 11.22%, 4.44%, and 39.94%, respectively; while the RMSE of BTGCN decreased by about 11.78% compared to the ARIMA model. Compared to

Table 3 Comparison of predictive performance across different methods on the data set

Time peried	Metric	HA	ARIMA	SVR	GCN	GRU	BiLSTM	BTGCN
24 h	MAE	3.1224	3.6155	2.5727	4.8725	3.2097	3.4551	2.1714
	RMSE	6.2081	4.3993	5.2243	8.6062	6.0434	5.9220	3.8809
	R^2	0.5245	*	0.6633	0.0866	0.5494	0.5277	0.8145
12 h	MAE	7.0186	7.9897	5.3829	9.8020	5.7839	6.1564	4.9502
	RMSE	13.1272	10.1689	10.2371	16.4343	11.1195	11.0783	8.3694
	R^2	0.4453	*	0.6626	0.1297	0.6018	0.5954	0.7744

SVR, BTGCN's RMSE and MAE decreased by approximately 25.71% and 15.60%, respectively. These significant performance improvements are attributed to the limitations of methods like HA, ARIMA, and SVR in handling complex non-stationary time series. The predictive performance of the GCN model is relatively lower, mainly because it only considers spatial features while neglecting the complexity of time series data. This is particularly prominent in task execution data, which comprises inherently typical time series data. In contrast, as a mature but relatively simple method, HA's predictive accuracy is usually lower than SVR's, mainly due to HA's limitations in handling long-term or non-stationary data, while SVR can address nonlinear issues through kernel tricks;

2) Strong spatial predictive capability. To assess the BTGCN model's ability to capture spatial features of system data, we conduct a comparative study with the GCN model, as shown in Fig. 7. Specifically, for the 24-h bottleneck prediction, the BTGCN model—compared to the GCN model which focuses solely on spatial features—demonstrates a reduction in RMSE by approximately 54.91% and also shows a significant decrease of 49.07% in the 12-h prediction, highlighting its remarkable capability in capturing spatial dependencies; and

3) Strong temporal predictive capability. To evaluate the BTGCN model's ability to capture the temporal features of system data, we conduct a comparative study with the GRU and BiLSTM models. The results indicate that the BTGCN model, which integrates spatio-temporal features, displays superior predictive accuracy compared to single-dimensional temporal feature models (GRU, BiLSTM), as shown in Fig. 7. This outcome validates the effectiveness of BTGCN in integrating and utilizing the temporal features of system data.

Compared to the GRU and BiLSTM models, which focus solely on temporal dynamics, the BTGCN model achieves a reduction in RMSE of 35.78% and 34.47%, respectively, in the 24-h bottleneck prediction, further proving its advantage in capturing time series correlations. Overall, the comparative results highlight the BTGCN model's superior capability in understanding and predicting complex system bottlenecks, emphasizing the importance of temporal feature data in accurate prediction.

4.6 Model interpretation

To better understand the BTGCN model, we select three tasks and visualize their prediction results on the test set. Figures 8, 9, and 10 display the visualization results for the throughput predictions of Actual Dispatch of Materials, Direct Issue Orders, and Material Sales Orders, respectively. In these visualizations, the horizontal axis represents the day number, while the vertical axis denotes the throughput of the tasks. In analyzing these results, we observe three key points:

1) The BTGCN model exhibits certain deviations in predicting the extremities (local minima/maxima) of various tasks (Figs. 8 and 9). This may stem from two factors: (a) GCN might tend toward smoothing in capturing spatial features—which in the Fourier domain, involves aggregating information from neighboring nodes to obtain a node's feature representation—thus aiding the identification of broad patterns and trends. Yet this smoothing operation could also lead to reduced sensitivity

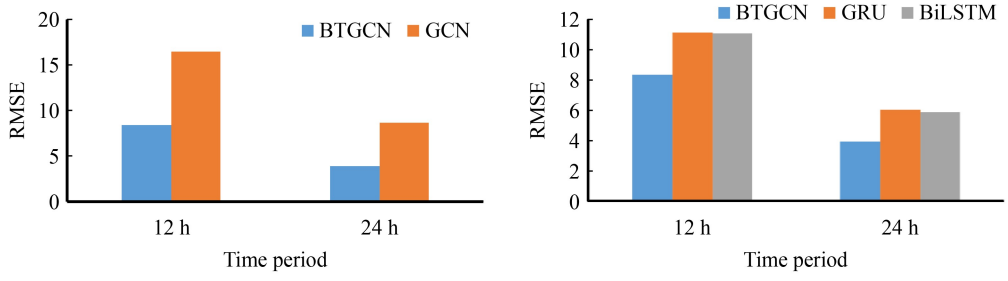

Fig. 7 Comparison of spatial and temporal predictive performance of BTGCN.

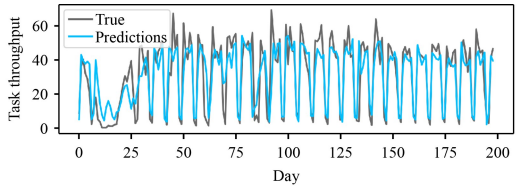

Fig. 8 Predicted throughput results for actual dispatch of materials.

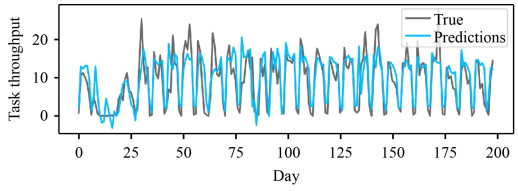

Fig. 9 Predicted throughput results for direct issue orders.

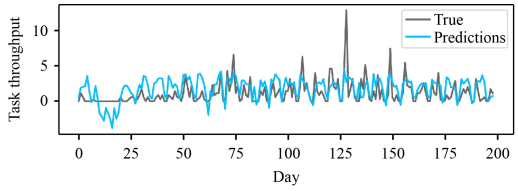

Fig. 10 Predicted throughput results for material sales orders.

of the model in capturing rapid changes or peaks, resulting in deviations at extremities; (b) Insufficient dynamism in BTGCN's handling of time series. Extremities in time series data are often caused by sudden events or irregular changes (such as holidays), requiring the model to have sufficient flexibility and sensitivity to capture these rapid changes;

2) For tasks with zero execution frequency, there is a certain discrepancy between the model predictions and the actual throughput information (Fig. 10). Zero frequency indicates that the task was not executed during certain periods, leading to data sparsity and causing the model to lack sufficient contextual information for accurate prediction; and

3) The BTGCN model demonstrates good overall adaptability and predictive accuracy for tasks with different characteristics. For example, it effectively captures the spatio-temporal features of task throughput, revealing trends in throughput changes. Notably, the model is capable of identifying the start and end of peak times, making the predictive pattern resemble the actual task execution frequency trend. These characteristics significantly enhance the model's ability to predict potential congestion and bottlenecks.

4.7 Validation of model generalizability and model limitations

To validate the generalizability of our BTGCN model and its applicability across a broader range of scenarios, we select ERP task operation data from a technology-intensive ship-building enterprise in China that significantly differs from the previously studied discrete ship-building enterprises, in that it has a much lower frequency of task execution. By comparing our model's performance on two distinct types of manufacturing enterprises—those with high task frequency versus those with low task frequency, as shown in Table 4 —we aim to assess its robustness and identify any industry-specific adjustments that may be required to do what.

BTGCN is designed to handle complex temporal and spatial dependencies in manufacturing processes. While it has shown significant advantages in discrete manufacturing enterprises with high-frequency task execution, its performance in technology-intensive enterprises, where task execution frequency is lower, is less impressive compared to traditional models like SVR. The primary reason for this disparity is the lower frequency of task executions in technology-intensive enterprises, which results in sparse temporal patterns. BTGCN relies on ample data points to effectively capture trends and dependencies over time, but when tasks are less frequent, the model struggles to find meaningful patterns, leading to reduced predictive accuracy. Additionally, the complexity of BTGCN makes it prone to overfitting in environments with limited data, whereas SVR, with its simpler regression approach, can handle sparse data more robustly and generalize better in these scenarios.

Moreover, SVR does not explicitly rely on temporal or spatial dependencies, allowing it to perform more consistently in diverse environments with varying data densities. The computational efficiency of SVR also makes it more suitable for scenarios with limited and infrequent data, where BTGCN's high computational requirements and dependency on extensive data collection can become a disadvantage. Hence, while BTGCN is highly effective in manufacturing environments with high task frequencies, SVR proves to be a more adaptable and reliable model in technology-intensive enterprises with lower task execution frequencies, highlighting the need for careful model selection based on the specific characteristics of the manufacturing process.

By incorporating data from different types of manufacturing enterprises, we find that while the BTGCN model

Table 4 Experimental results of BTGCN model on the new data set

Metric	HA	ARIMA	SVR	GCN	GRU	BiLSTM	BTGCN
MAE	48.2023	27.8615	32.8471	61.6747	33.5278	38.2815	43.0337
RMSE	64.0112	35.5261	48.3570	77.8279	49.8910	47.0917	54.7854
R^2	0.4799	*	0.6724	0.1137	0.6450	0.6102	0.5618

is adaptable to varying conditions, its efficacy is highly dependent on the nature and frequency of task executions. Future research could explore modifications to enhance the model's performance in environments with lower task frequencies, ensuring broader applicability across diverse manufacturing sectors.

5 Discussion

5.1 Scalability and adaptability of the model

In the rapidly evolving manufacturing environment, changes in operational processes and production scales demand higher adaptability from predictive models. To ensure the applicability of the BTGCN model across various manufacturing scenarios, we need to delve deeper into its scalability and adaptability. This exploration requires a theoretical analysis of the model's structure to predict its performance under different conditions.

The BTGCN model integrates GCN and GRU, where GCN handles spatial features by parsing the topological structure of the task network, whereas GRU processes temporal data to capture dynamic changes over time. This design provides the BTGCN model with substantial flexibility in managing complex manufacturing task networks. To adapt to different manufacturing processes, the BTGCN model must select features that are pertinent to the specific application scenario. For instance, in discrete manufacturing, task dependencies and execution sequences are crucial, while in process manufacturing, resource utilization and production efficiency might be the key operational features. Feature selection should thus be tailored to the specific manufacturing process to maximize the model's predictive accuracy.

The scalability of the BTGCN model is reflected in its ability to handle varying data scales. As the data scale increases, model parameters (such as the number of GCN layers and GRU units) need to be adjusted accordingly to maintain performance. Computational complexity and training time are also critical factors affecting the model's scalability. For large-scale data sets, the BTGCN model may require significant computational resources, necessitating optimizations in model architecture or the use of distributed computing techniques to enhance training efficiency.

5.2 Integration into existing manufacturing systems

The BTGCN model's ability to integrate into existing manufacturing systems and its potentially significant contribution to strategic decision-making can significantly enhance manufacturing efficiency. Specifically, the model can be incorporated into a supply chain management system to predict potential bottlenecks and optimize logistics. For instance, in a car manufacturing company, the supply chain is complex, involving multiple suppliers and assembly lines. By predicting bottlenecks, the BTGCN model can enable proactive adjustments in the supply chain and improve logistical efficiency, thus ensuring timely delivery of parts and materials. Nevertheless, there are several challenges in integrating the BTGCN model into existing systems, which can include data collection and integration, model training, real-time processing, interdepartmental coordination, and cost-benefit analysis. First of all, while it can be difficult to incorporate data from various suppliers and internal systems, this can be addressed by implementing a unified data platform that effectively consolidates data from all sources, ensuring consistency and availability. While predicting bottlenecks in real-time requires substantial computational power, this can be managed through the use of cloud computing and distributed processing to handle large-scale, real-time data analytics.

As effective use of bottleneck predictions also requires coordination among departments, developing a centralized dashboard that provides actionable insights to all relevant departments can enhance communication and collaboration. For example, in a semiconductor manufacturing plant, the BTGCN model can be integrated into the predictive maintenance framework to proactively forecast machine failures and schedule maintenance. By identifying when and where failures are likely to occur, the model enables managers to address issues before they escalate, thus preventing unplanned downtime and maintaining continuous production. This integration can lead to a more efficient use of resources, as maintenance activities can be scheduled during planned downtimes rather than being limited to reacting to unexpected failures. The integration process can thus minimize downtime and improve overall equipment effectiveness (OEE).

Moreover, accurate prediction requires extensive data from various sensors and machines, which can be addressed by implementing IoT devices for comprehensive data collection and ensuring regular maintenance of these devices to avoid data gaps. Training the model with historical data to recognize failure patterns can be time-consuming, but this issue can be mitigated by using incremental learning techniques to continuously update the model with new data, reducing the initial training time. Justifying the investment in predictive maintenance tools can also be difficult but conducting a pilot project to demonstrate the potential return on investment (ROI)—focusing on reduced downtime and maintenance costs—can help illustrate the benefits.

5.2.1 Advantages in strategic decision-making

The strategic use of the BTGCN model can extend beyond immediate operational improvements. By providing accurate predictions and insights, the model can support long-term strategic decision-making. For example, it

can help in strategic resource planning, determining optimal investment in machinery and workforce based on predicted production trends. Insights delivered by the model can also guide decisions on product development and market strategy by identifying potential production constraints and enabling proactive measures to address them. Overall, the BTGCN model's ability to provide actionable data can lead to more informed and effective strategic decisions, ultimately enhancing the company's competitive advantage in the market.

6 Conclusions and future directions

6.1 Conclusions

Extensive research on task bottlenecks has confirmed how these constraints significantly impact output and process timing, making the identification and prediction of these bottlenecks a crucial task in enterprise management (Li et al., 2009). In business environments where bottleneck locations are relatively fixed, accurately detecting bottleneck positions is particularly critical. In dynamic environments where bottleneck positions frequently change, the ability to effectively predict the emergence and variation of bottlenecks becomes even more important, as it can help managers take measures in advance that can reduce the negative impact of bottlenecks on system efficiency. Against these dynamically changing backdrops, bottleneck prediction necessitates the consideration of both temporal and spatial features. Addressing this issue, this study focuses on task bottleneck prediction in digital manufacturing enterprises using the novel BTGCN model, which is capable of effectively extracting the spatio-temporal features between tasks. This proposed model employs a spatial GCN neural network to capture the spatial relationships between tasks and integrates these spatial features with the temporal characteristics of corresponding time periods. This approach— distinct from the stacking mechanisms adopted in traditional literature—more effectively merges spatial and temporal data. Experimental results demonstrate that the proposed BTGCN model surpasses other baseline models in predictive performance.

6.2 Limitations

This study presents several limitations related to the BTGCN model and the data set we use to evaluate its properties. Model limitations include: (1) challenges in real-time data processing, because BTGCN requires substantial computational resources and time, which can lead to processing delays and thus affect the timely prediction of bottlenecks; (2) The BTGCN model is more suited for discrete manufacturing enterprises. It struggles to achieve effective predictions for technology-intensive enterprises with lower task execution frequencies; (3) The accuracy of the model is highly dependent on the quality of input data. Incomplete or inaccurate data can significantly degrade its performance, highlighting the importance of robust data collection and pre-processing methods; and (4) As the attributes of different tasks may affect throughput, the impact of task attribute features is overlooked in the model. Future research can explore incorporating these features into the model to improve prediction accuracy. Data set limitations include: (1) Focus on the ship-building industry, which may limit the applicability of the research findings to other manufacturing industries. Expanding the scope of the study to cover more industries will help enhance the model's generalizability; and (2) The integration of spatial and temporal data may introduce biases, especially if the data distribution is uneven or there are inherent patterns that the model cannot adequately capture. This can affect the model's applicability in different scenarios. In summary, these limitations indicate that future research needs to improve both the model and the data aspects to enhance the applicability and prediction accuracy of the BTGCN model in various manufacturing environments.

6.3 Future research directions

To provide valuable guidance for future research, this study suggests specific methodological improvements and new application directions. First, exploring machine learning algorithms for dynamic adaptation of task networks can significantly enhance the model's flexibility and responsiveness to ever-changing manufacturing environments. For example, integrating reinforcement learning techniques can enable the BTGCN model to dynamically adjust its parameters based on real-time feedback from the manufacturing environment, thereby improving prediction accuracy. Additionally, investigating the application of the BTGCN model in other digital manufacturing domains—such as predictive maintenance and supply chain optimization—can broaden its practical applications. Employing the BTGCN model to predict equipment failures and optimize maintenance schedules can thus help reduce downtime and increase overall equipment efficiency. Similarly, leveraging this model to enhance supply chain logistics by predicting and mitigating potential disruptions can lead to more resilient and efficient manufacturing operations.

Moreover, improving the model's scalability and computational efficiency is also a promising area for future research. Techniques such as model pruning, quantization, and distributed computing can be explored to reduce the computational burden, making the BTGCN model more suitable for real-time applications in large-scale manufacturing environments. Finally, conducting comprehensive case studies and practical experiments

across various manufacturing industries is essential for validating the model's effectiveness and identifying any necessary industry-specific adjustments. Collaboration with industry partners could facilitate the collection of diverse data sets and provide practical insights into the model's implementation and performance in different contexts. At the same time, with the rapid development of Industry 4.0 and Industry 5.0, the deep integration of lean production principles and smart manufacturing has further enhanced system efficiency and resource optimization (Hasan and Trianni, 2023), offering more possibilities for the application of the BTGCN model in complex manufacturing environments. Future research should therefore focus on methodological advancements and new applications of the BTGCN model, aiming to enhance its adaptability, scalability, and practical utility in the rapidly evolving field of digital manufacturing.

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

References

Ahuett-Garza H, Kurfess T (2018). A brief discussion on the trends of habilitating technologies for Industry 4.0 and Smart manufacturing. Manufacturing Letters, 15: 60–63

Arjovsky M, Shah A, Bengio Y (2016). Unitary evolution recurrent neural networks. In: Proceedings of The 33rd International Conference on Machine Learning, PMLR 48:1120-1128

Biller S, Jingshan Li, Marin S P, Meerkov S M, Liang Zhang (2010). Bottlenecks in Bernoulli serial lines with rework. IEEE Transactions on Automation Science and Engineering, 7(2): 208–217

Bronstein M M, Bruna J, Lecun Y, Szlam A, Vandergheynst P (2017). Geometric deep learning: Going beyond euclidean data. IEEE Signal Processing Magazine, 34(4): 18–42

Buffelli D, Vandin F (2022). The impact of global structural information in graph neural networks applications. Data, 7(10): 1–20

Cao Z, Deng J, Liu M, Wang Y (2012). Bottleneck prediction method based on improved adaptive network-based fuzzy inference system (ANFIS) in Semiconductor Manufacturing System. Chinese Journal of Chemical Engineering, 20(6): 1081–1088

Chang X, Jia X (2023). A DeepAR based hybrid probabilistic prediction model for production bottleneck of flexible shop-floor in Industry 4.0. Computers & Industrial Engineering, 185: 109644

Chryssolouris G, Mavrikios D, Papakostas N, Mourtzis D, Michalos G, Georgoulias K (2009). Digital manufacturing: history, perspectives, and outlook. Proceedings of the Institution of Mechanical Engineers. Part B, Journal of Engineering Manufacture, 223(5): 451–462

Fang W, Guo Y, Liao W, Huang S, Yang N, Liu J (2020). A Parallel Gated Recurrent Units (P-GRUs) network for the shifting lateness bottleneck prediction in make-to-order production system. Computers & Industrial Engineering, 140: 106246

Fionda V, Gutierrez C, Pirrò G (2016). Building knowledge maps of Web graphs. Artificial Intelligence, 239: 143–167

Goldratt E M, Cox J (1984). The goal: A process of ongoing improvement. Taylor & Francis

Hajlaoui R, Moulahi T, Zidi S, El Khediri S, Alaya B, Zeadally S (2024). Towards smarter cyberthreats detection model for industrial internet of things (IIoT) 4.0. Journal of Industrial Information Integration, 39: 100595

Hasan A S M M, Trianni A (2023). Boosting the adoption of industrial energy efficiency measures through Industry 4.0 technologies to improve operational performance. Journal of Cleaner Production, 425: 138597

Hu W, Fey M, Zitnik M, Dong Y, Ren H, Liu B, Catasta M, Leskovec J (2020). Open graph benchmark: Datasets for machine learning on graphs. In: Advances in Neural Information Processing Systems, 33: 22118–22133

Huang S Y, Chen H J, Chiu A A, Chen C P (2014). The application of the theory of constraints and activity-based costing to business excellence: The case of automotive electronics manufacture firms. Total Quality Management & Business Excellence, 25(5–6): 532–545

Kiggundu M N (1981). Task interdependence and the theory of job design. Academy of Management Review, 6(3): 499–508

Kisvari A, Lin Z, Liu X (2021). Wind power forecasting—A data-driven method along with gated recurrent neural network. Renewable Energy, 163: 1895–1909

Krizhevsky A, Sutskever I, Hinton G E (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6): 84–90

Lai X, Shui H, Ding D, Ni J (2021). Data-driven dynamic bottleneck detection in complex manufacturing systems. Journal of Manufacturing Systems, 60: 662–675

Lai X, Shui H, Ni J (2018). A two-layer long short-term memory network for bottleneck prediction in multi-job manufacturing systems. In: ASME 2018 13th International Manufacturing Science and Engineering Conference, 3

Li G, Müller M, Qian G, Delgadillo I C, Abualshour A, Thabet A, Ghanem B (2023a). DeepGCNs: Making GCNs go as deep as CNNs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(6): 6923–6939

Li L (2018). A systematic-theoretic analysis of data-driven throughput bottleneck detection of production systems. Journal of Manufacturing Systems, 47: 43–52

Li L, Chang Q, Ni J (2009). Data driven bottleneck detection of manufacturing systems. International Journal of Production Research, 47(18): 5019–5036

Li L, Chang Q, Xiao G, Ambani S (2011). Throughput bottleneck prediction of manufacturing systems using time series analysis. Journal of Manufacturing Science and Engineering, 133(2): 021015

Li X, Sun L, Ling M, Peng Y (2023b). A survey of graph neural network based recommendation in social networks. Neurocomputing, 549: 126441

Monti F, Boscaini D, Masci J, Rodola E, Svoboda J, Bronstein M M (2017). Geometric deep learning on graphs and manifolds using mixture model cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR): 5115–5124

Parisot S, Ktena S I, Ferrante E, Lee M, Guerrero R, Glocker B, Rueckert D (2018). Disease prediction using graph convolutional

networks: application to autism spectrum disorder and Alzheimer's disease. Medical Image Analysis, 48: 117–130

Peng H, Zhang R, Dou Y, Yang R, Zhang J, Yu P S (2022). Reinforced neighborhood selection guided multi-relational graph neural networks. ACM Transactions on Information Systems, 40(4): 1–46

Piccarozzi M, Silvestri L, Silvestri C, Ruggieri A (2024). Roadmap to Industry 5.0: Enabling technologies, challenges, and opportunities towards a holistic definition in management studies. Technological Forecasting and Social Change, 205: 123467

R. Thorne D (2006). Throughput: a simple performance index with desirable characteristics. Behavior Research Methods, 38(4): 569–573

Rezvanian A, Meybodi M R (2016). Stochastic graph as a model for social networks. Computers in Human Behavior, 64: 621–640

Shen G, Tan Q, Zhang H, Zeng P, Xu J (2018). Deep learning with gated recurrent unit networks for financial sequence predictions. Procedia Computer Science, 131: 895–903

Subramaniyan M, Skoogh A, Muhammad A S, Bokrantz J, Johansson B, Roser C (2020). A generic hierarchical clustering approach for detecting bottlenecks in manufacturing. Journal of Manufacturing Systems, 55: 143–158

Subramaniyan M, Skoogh A, Salomonsson H, Bangalore P, Bokrantz J (2018a). A data-driven algorithm to predict throughput bottlenecks in a production system based on active periods of the machines. Computers & Industrial Engineering, 125: 533–544

Subramaniyan M, Skoogh A, Salomonsson H, Bangalore P, Gopalakrishnan M, Sheikh Muhammad A (2018b). Data-driven algorithm for throughput bottleneck analysis of production systems. Production & Manufacturing Research, 6(1): 225–246

Sun B, Zhao D, Shi X, He Y (2021). Modeling global spatial–temporal graph attention network for traffic prediction. IEEE Access: Practical Innovations, Open Solutions, 9: 8581–8594

Tallec C, Ollivier Y (2018). Can recurrent neural networks warp time? arXiv:1804.11188

Vashishth S, Sanyal S, Nitin V, Talukdar P (2019). Composition-based multi-relational graph convolutional networks. arXiv:1911.03082

Wu Z, Meng Z, Gray J (2017). IoT-based techniques for online M2M-interactive itemized data registration and offline information traceability in a digital manufacturing system. IEEE Transactions on Industrial Informatics, 13(5): 2397–2405

Wu Z, Pan S, Chen F, Long G, Zhang C, Yu P S (2021). A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems, 32(1): 4–24

Xiang W, Yu K, Han F, Fang L, He D, Han Q L (2024). Advanced manufacturing in industry 5.0: A survey of key enabling technologies and future trends. IEEE Transactions on Industrial Informatics, 20(2): 1055–1068

Ye Z, Kumar Y J, Sing G O, Song F, Wang J (2022). A comprehensive survey of graph neural networks for knowledge graphs. IEEE Access: Practical Innovations, Open Solutions, 10: 75729–75741

Yu C, Matta A (2016). A statistical framework of data-driven bottleneck identification in manufacturing systems. International Journal of Production Research, 54(21): 6317–6332

Zhang D, Lindholm G, Ratnaweera H (2018). Use long short-term memory to enhance Internet of Things for combined sewer overflow monitoring. Journal of Hydrology, 556: 409–418

Zhang T, Shan H R, Little M A (2022). Causal GraphSAGE: A robust graph method for classification based on causal sampling. Pattern Recognition, 128: 108696

Zhang Y, Pal S, Coates M, Ustebay D (2019). Bayesian graph convolutional neural networks for semi-supervised classification. In: Proceedings of The 33rd AAAI Conference on Artificial Intelligence (AAAI-19):5829–5836

Zhao F, Xia S, Wu Z, Duan D, Wang L, Lin W, Gilmore J H, Shen D, Li G (2019). Spherical U-Net on cortical surfaces: Methods and applications. In: Chung A, Gee J, Yushkevich P, Bao S, eds. Information Processing in Medical Imaging. IPMI 2019. Lecture Notes in Computer Science, 11492: 855–866