Jurnal Teknik Informatika C.I.T Medicom, 16 (2) (2024) 90-102

Published by: Institute of Computer Science (IOCS)

Jurnal Teknik Informatika C.I.T Medicom

Journal homepage: www.medikom.iocspublisher.org

Advanced graph neural networks for dynamic yield optimization and resource allocation in industrial systems

Lise Pujiastuti[1], and Mochamad Wahyudi[2]

[1)] Program Studi Sistem Informasi, STMIK Antar Bangsa, Tangerang, Banten, Indonesia

[2)] Program Studi Informatika, Universitas Bina Sarana Infotmatika, Jakarta, DKI Jakarta, Indonesia

Article Info

Article history

Received : Mar 10, 2024
Revised : Apr 19, 2024
Accepted : May 28, 2024

Keywords:

Dynamic Yield Optimization;
Graph Neural Networks (GNNs);
Industrial Systems;
Reinforcement Learning (RL);
Resource Allocation.

Abstract

This research explores the integration of Graph Neural Networks (GNNs) and Reinforcement Learning (RL) for dynamic yield optimization and resource allocation in industrial systems. We present a numerical example involving a small manufacturing setup with three machines, where GNNs are employed to model complex interactions and derive meaningful embeddings of machine states. These embeddings are then used to predict yield and cost through linear combination functions. RL is utilized to optimize resource allocation dynamically, balancing yield and cost through a carefully designed reward function. The results demonstrate the effectiveness of GNNs in capturing machine interactions and the adaptability of RL in optimizing operational parameters in real-time. This combined approach showcases significant potential for enhancing efficiency, cost-effectiveness, and overall performance in various industrial applications, providing a robust framework for continuous improvement and adaptive decision-making in dynamic environments.

Corresponding Author:

Lise Pujiastuti,
Program Studi Sistem Informasi,
STMIK Antar Bangsa, Tangerang ,
Kawasan Bisnis CBD Ciledug, Jl. HOS Cokroaminoto No.29-35 Blok A5, RT.001/RW.001, Banten 15157, Indonesia,
Email: lise.pujiastuti@gmail.com

1. Introduction

Industrial systems are increasingly becoming complex and data-driven, necessitating advanced methods for optimizing yield and allocating resources efficiently[1]–[4]. Yield optimization and resource allocation are pivotal for enhancing productivity and reducing costs in industrial operations[5]. Traditional approaches often fail to dynamically adapt to the ever-changing industrial environments, resulting in inefficiencies and increased waste[6]. Graph Neural Networks (GNNs), with their unique ability to model complex relationships and dependencies, offer a promising solution to these challenges[7], [8]. This research aims to explore the application of advanced GNN techniques to achieve dynamic yield optimization and efficient resource allocation in industrial systems[9], [10].

Graph Neural Networks (GNNs) have emerged as powerful tools for inference on graph-structured data[11], [12]. They excel at capturing the intricate dependencies and relationships between entities represented as nodes and their interactions as edges[13], [14]. In industrial contexts, processes can be naturally represented as graphs, where machines, sensors, and products are nodes, and their interactions form the edges[15], [16]. Yield optimization involves maximizing the output quality and quantity from industrial processes, while resource allocation ensures the efficient distribution of

materials, energy, and labor. Both tasks are dynamic and require real-time adaptation to changing conditions, making them ideal candidates for GNN-based approaches.

The dynamic nature of industrial systems poses significant challenges for yield optimization and resource allocation[17]–[19]. Traditional methods, relying on static models and fixed parameters, are inadequate for handling the continuous fluctuations in operational conditions[20], [21]. Changes in raw material quality, machine performance, and environmental factors necessitate dynamic adjustment of process parameters, which current methods struggle to achieve[22]. Furthermore, existing resource allocation strategies often lack the flexibility to adapt to varying demands and conditions in real-time, leading to inefficiencies and bottlenecks[23].

Recent studies have demonstrated the potential of GNNs in various domains, including social networks, recommendation systems, and biological networks[24], [25]. These studies have shown that GNNs can effectively model complex, dynamic systems and make accurate predictions based on the relationships and dependencies within the data[26]. In the context of industrial systems, preliminary research has indicated that GNNs can be employed to optimize processes and improve decision-making[27]. However, there is a lack of comprehensive studies focusing specifically on dynamic yield optimization and resource allocation using advanced GNN techniques.

The theoretical foundation of GNNs lies in their ability to perform convolution operations on graphs, similar to how Convolutional Neural Networks (CNNs) operate on grid-structured data[8]. GNNs aggregate and propagate information across the graph, capturing both local and global structures[28]. Variants like Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and GraphSAGE enhance this capability by introducing mechanisms to weigh the importance of nodes and edges and enabling inductive learning for generalization to unseen data[29]. These properties make GNNs particularly suitable for modeling the dynamic and complex nature of industrial systems[30], [31].

The research will be conducted in several phases[32]. Initially, industrial systems will be modeled as graphs, capturing the components and their interactions[33]. Advanced GNN architectures, including GCNs, GATs, and GraphSAGE, will be developed to handle large-scale, dynamic graphs[34]. Reinforcement learning algorithms will be integrated to enable real-time adjustment of process parameters for yield optimization. Multi-agent systems will be employed to develop efficient resource allocation strategies. Extensive testing will be conducted in both simulated environments and real-world industrial settings to validate the models. Finally, the GNN models will be integrated with industrial IoT systems for real-time data processing and continuous learning.

The primary objective of this research is to develop advanced GNN models that can dynamically optimize yield and allocate resources efficiently in industrial systems. By leveraging the power of GNNs, the research aims to achieve significant improvements in yield and resource utilization, enhancing overall efficiency and reducing costs. The developed models will be scalable, capable of handling large-scale industrial systems with complex interactions. Additionally, the real-time adaptability of the models will ensure continuous optimization, allowing industrial systems to respond effectively to changing conditions. The integration with industrial IoT systems will further enable seamless data collection and processing, fostering a data-driven approach to industrial management.

This research has the potential to revolutionize industrial operations by providing advanced tools for dynamic yield optimization and resource allocation. The anticipated benefits include enhanced productivity, reduced waste, and lower operational costs, contributing to more sustainable and efficient industrial practices.

2. Research Methods

In this section we explain how to complete this research to answer the research problem that has been outlined in the introduction[35], [36].

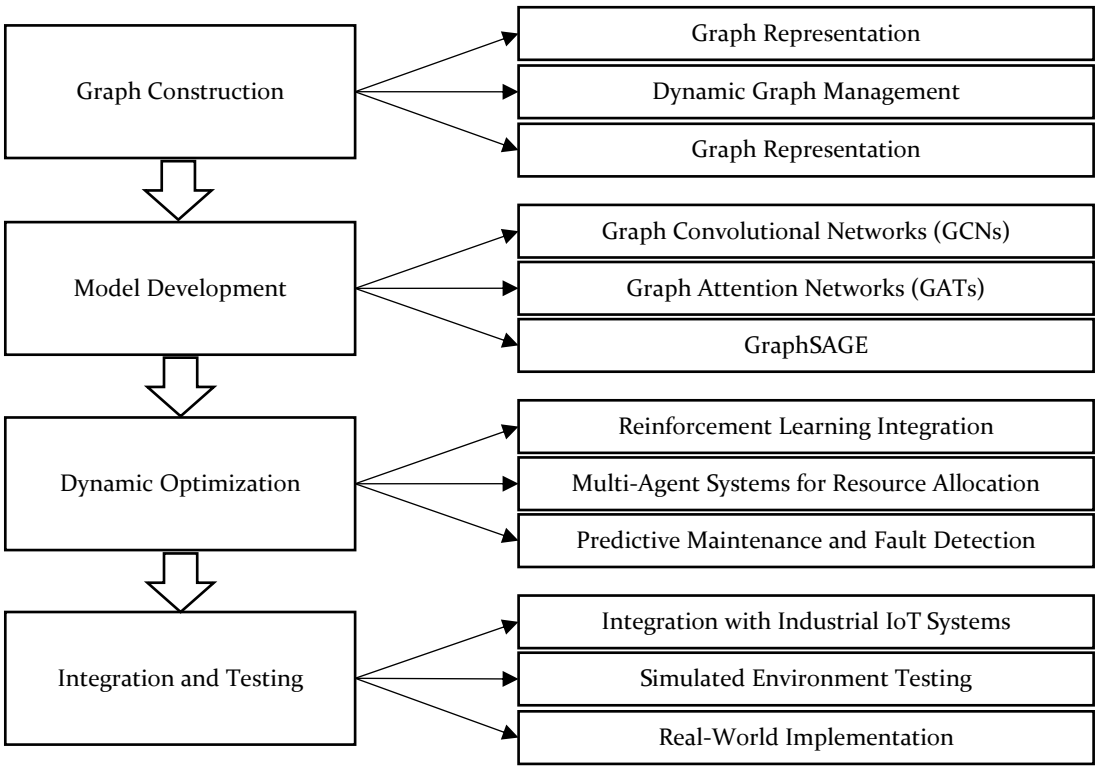

Figure 1. Model Development flow[37]

Below is an explanation of Figure 1 above:

1) Graph Construction
   a) Data Collection:
      The first step involves collecting data from various components of the industrial system, including machines, sensors, and production lines. This data encompasses operational parameters, process metrics, and interaction logs.
   b) Graph Representation:
      The industrial system will be modeled as a dynamic graph where:
      - Nodes represent components such as machines, sensors, and products.
      - Edges represent interactions or dependencies, such as material flow, energy transfer, and communication links.
   c) Dynamic Graph Management:
      Since industrial systems are dynamic, the graph representation will continuously update to reflect changes over time, capturing temporal variations in the system's state.
2) Model Development
   a) Graph Convolutional Networks (GCNs):
      GCNs will be employed to aggregate and propagate information across the graph. This helps in understanding the local structure of the industrial system and the direct interactions between components.
   b) Graph Attention Networks (GATs):
      GATs will be used to weigh the importance of different nodes and edges dynamically. This enables the model to focus on critical components and interactions that significantly impact yield and resource allocation.
   c) GraphSAGE:

GraphSAGE will be implemented for inductive learning, allowing the model to generalize and make predictions for previously unseen parts of the graph. This is crucial for handling new components or processes introduced in the industrial system.

3) Dynamic Optimization
   a) Reinforcement Learning Integration:
   Reinforcement learning (RL) algorithms will be integrated with the GNN models to enable dynamic adjustment of process parameters. The RL agents will learn to optimize yield by receiving feedback from the system's performance and adjusting actions accordingly.
   b) Multi-Agent Systems for Resource Allocation:
   Multi-agent systems will be developed to manage resource allocation. Each agent will represent a different resource type (e.g., materials, energy, labor) and will coordinate with other agents to ensure optimal distribution based on real-time data and system demands.
   c) Predictive Maintenance and Fault Detection:
   The GNN models will incorporate predictive maintenance algorithms to identify potential faults and failures before they occur. This will involve analyzing patterns in the graph data to predict when and where maintenance is needed, reducing downtime and improving system reliability.
4) Integration and Testing
   a) Integration with Industrial IoT Systems:
   The developed GNN models will be integrated with existing industrial IoT systems for seamless data collection, processing, and real-time decision-making. This integration ensures that the models can leverage the continuous stream of data from sensors and devices in the industrial environment.
   b) Simulated Environment Testing:
   Initial testing will be conducted in a simulated industrial environment to validate the models' performance and effectiveness. This will allow for controlled experimentation and fine-tuning of the models before deployment.
   c) Real-World Implementation:
   After successful validation in simulation, the models will be deployed in real-world industrial settings. Continuous monitoring and evaluation will be performed to assess the models' impact on yield optimization and resource allocation.
   d) Continuous Learning and Adaptation:
   The models will be designed to learn continuously from the real-time data, adapting to changes in the industrial system. This ensures that the optimization strategies remain effective even as the system evolves.

By following this comprehensive methodology, the research aims to develop robust and scalable GNN-based models that can dynamically optimize yield and allocate resources efficiently in industrial systems. The integration with industrial IoT systems will enable real-time data-driven decision-making, leading to significant improvements in productivity, efficiency, and cost reduction.

Introduction to Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) are a class of neural networks designed to operate on graph-structured data[38], [39]. They are particularly effective at modeling relational data and capturing the dependencies between interconnected entities[40], [41]. GNNs propagate and transform feature information across nodes and edges, making them well-suited for applications in industrial systems where processes and components are highly interconnected.

GNNs in Industrial Systems

In an industrial setting, we can represent the entire system as a grap $G = (V, E)$, where:

- V is the set of nodes, representing components such as machines, sensors, and products.

- E is the set of edges, representing interactions such as material flows, energy transfers, and communication links.

The goal is to utilize GNNs to dynamically optimize yield and allocate resources efficiently by leveraging real-time data and the complex dependencies within the system.

Basic Mathematical Formulation

Node Features and Edge Features:

- Each node $v \in V$, has a feature vector then $\mathbf{h}_v$.
- Each edge $e = (u, v) \in E$, has a feature vector $\mathbf{e}_{uv}$.

Graph Convolutional Network (GCN):

A GCN updates the node features by aggregating information from neighboring nodes. The basic operation of a GCN layer can be described as:

$$\mathbf{h}_v^{(k-1)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \frac{1}{c_{vu}} \boldsymbol{w}^{(k)} \mathbf{h}_u^k + \mathbf{w}_0^k \, \mathbf{h}_u^k\right) \quad (1)$$

where:

- $\mathbf{h}_u^k$ is the feature vector of node v at layer k.
- $\mathcal{N}(v)$ denotes the set of neighbors of node v.
- c_{vu} is a normalization constant, often chosen as $\sqrt{|\mathcal{N}(v)||\mathcal{N}(u)|}$.
- $\boldsymbol{w}^{(k)}$and $\mathbf{w}_0^k$ are learnable weight matrices.
- σ is an activation function, such as ReLU.

Graph Attention Network (GAT):

A GAT layer introduces attention mechanisms to weigh the importance of neighboring nodes:

$$\mathbf{h}_v^{(k-1)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \boldsymbol{\alpha}_{vu}^{(k)} \, \boldsymbol{w}^{(k)} \mathbf{h}_u^k\right) \quad (2)$$

where:

$\boldsymbol{\alpha}_{vu}^{(k)}$ are attention coefficients computed as: $\boldsymbol{\alpha}_{vu}^{(k)} = \frac{\exp\left(\text{LeakyReLU}\left(\text{a}^T \left|\mathbf{w}^{(k)}\mathbf{h}_v^{(k)}\right|\left|\mathbf{w}^{(k)}\mathbf{h}_u^{(k)}\right|\right)\right)}{\sum_{j \in \mathcal{N}(v)} \exp\left(\text{LeakyReLU}\left(\text{a}^T \left|\mathbf{w}^{(k)}\mathbf{h}_v^{(k)}\right|\left|\mathbf{w}^{(k)}\mathbf{h}_j^{(k)}\right|\right)\right)}$

where a is a learnable weight vector and || denotes concatenation.

GraphSAGE:

GraphSAGE performs inductive node representation learning by sampling and aggregating features from a node's local neighborhood:

$$\mathbf{h}_v^{(k+1)} = \sigma\left(\boldsymbol{w}^{(k)}\left[\mathbf{h}_v^k || \text{AGGREGATE}^{(k)}(\{\mathbf{h}_u^k, \forall u \in \mathcal{N}(v)\})\right]\right) \quad (3)$$

Where $\text{AGGREGATE}^{(k)}$ is a function such as mean, LSTM, or pooling that combines the features of the neighbors.

Dynamic Yield Optimization

The objective of yield optimization is to maximize the output quality and quantity while minimizing waste. This can be formulated as a dynamic optimization problem where the GNN models predict the optimal process parameters based on the current state of the system graph.

Let y denote the yield, x the input parameters, and z the system state captured by the GNN. The optimization problem can be expressed as:

$$\max_x y(x, z) \quad (4)$$

subject to constraints imposed by the industrial process and resource availability.

Resource Allocation

Resource allocation involves distributing resources R (materials, energy, labor) to various nodes and edges to optimize the overall system performance. The GNN models help predict the optimal allocation strategy by learning the complex dependencies and interactions within the system.

Let r denote the resources to be allocated and c the cost function. The optimization problem can be formulated as:

$$\min_r c(r, z) \quad (5)$$

Integration with Reinforcement Learning

To enable dynamic optimization, reinforcement learning (RL) algorithms will be integrated with GNNs. The RL agent will learn to adjust process parameters x and resource allocations r by interacting with the industrial system environment and receiving feedback in the form of rewards R.

The objective is to maximize the cumulative reward over time, which corresponds to improved yield and efficient resource utilization:

$$\max \sum_t R_t\,(x_t, r_t, z_t) \tag{6}$$

where t denotes the time step.

By leveraging advanced GNN techniques and integrating them with RL algorithms, this research aims to develop robust and scalable models for dynamic yield optimization and resource allocation in industrial systems, leading to significant improvements in efficiency, productivity, and cost reduction.

Advanced Graph Neural Networks for Dynamic Yield Optimization and Resource Allocation in Industrial Systems.

Graph Representation of Industrial Systems

We represent the industrial system as a dynamic graph $\mathbf{G}_t = (\mathbf{V}_t, \mathbf{E}_t)$ at time t.

$\mathbf{V}_t$: Set of nodes representing components (machines, sensors, products).

$\mathbf{E}_t$: Set of edges representing interactions (material flows, energy transfers).

Each node $v \in \mathbf{V}_t$ has a feature vector $\mathbf{h}_v^{(t)}$ and each edge $e = (u, v) \in E_t$ has a feature vector $\mathbf{e}_{uv}^{(t)}$

Node and Edge Embeddings

We learn node and edge embeddings through multiple layers of a Graph Neural Network (GNN):

For a GNN layer k, the update rule for node v is:

$$\mathbf{h}_v^{(k-1),t} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \boldsymbol{\alpha}_{vu}^{(k),t}\, \boldsymbol{w}^{(k)} \mathbf{h}_u^{(k),t} + \mathbf{w}_0^k\, \mathbf{h}_u^{(k),t}\right) \tag{7}$$

Where $\boldsymbol{\alpha}_{vu}^{(k),t}$ are attention coefficients or normalization factors, $\boldsymbol{w}^k$ and $\mathbf{w}_0^k$ are learnable weight matrices, and σ is an activation function such as ReLU.

Dynamic Yield Optimization

The objective is to maximize yield y_t given input parameters x_t and the system state z_t (node and edge embeddings).

Optimization Problem:

$$\max_{x_t} y_t(x_t, z_t) \tag{8}$$

subject to operational constraints $g(x_t, z_t) \leq 0$

Yield Prediction:

Given the current state zt, the GNN predicts the yield:

$$y_x = fGNN(x_t, z_t) \tag{9}$$

Resource Allocation

The goal is to allocate resources r_t efficiently to optimize system performance.

Optimization Problem:

$$\min_{r_t} c_t(r_t, z_t) \tag{10}$$

subject to resource constraints $\mathbf{h}(r_t, z_t) = \mathbf{b}$

Resource Allocation Strategy:

The GNN models the optimal resource allocation:

$$y_x = gGNN(z_t, c_t) \tag{11}$$

Integration with Reinforcement Learning (RL)

We integrate reinforcement learning (RL) to dynamically optimize yield and resource allocation. An RL agent interacts with the industrial system, receiving state s_t and taking actions $\mathbf{a}_t$ to maximize cumulative rewards.

State Representation:

The state s_t is derived from the GNN embeddings $\mathbf{z}_t$.
Action Selection:
Actions a_t correspond to adjusting process parameters x_t and resource allocations z_t.
Reward Function:
The reward R_t reflects improvements in yield, efficiency, and cost reduction:

$$R_x = y_t - \lambda c_t \tag{12}$$

where λ is a weighting factor balancing yield and cost.
Objective:
The RL agent aims to maximize the cumulative reward over time:

$$\max \sum_{t=0}^{T} \gamma^t R_t(\mathbf{s}_t, \mathbf{a}_t) \tag{13}$$

where γ is a discount factor.
Policy Optimization:
The policy $\pi(\mathbf{a}_t|\mathbf{s}_t)$ is optimized using policy gradient methods or Q-learning:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi\theta}\left[\sum_{t0}^{T} \nabla_\theta \log \pi_\theta\,(\mathbf{a}_t|\mathbf{s}_t) R_t\right] \tag{14}$$

Combined Formulation
Combining the GNN and RL formulations, we have:

$$\mathbf{h}_v^{(k-1),t} = \sigma\left(\sum_{u\in\mathcal{N}(v)} \boldsymbol{\alpha}_{vu}^{(k),t}\, \boldsymbol{w}^{(k)} \mathbf{h}_u^{(k),t} + \mathbf{w}_0^k\, \mathbf{h}_u^{(k),t}\right)$$
$$\mathbf{z}_t = \left\{\mathbf{h}_v^{(L),t}\right\} v \in V_t \tag{15}$$

Yield Optimization:

$$\max_{x_t} f\mathrm{GNN}(\mathbf{z}_t, \mathbf{x}_t) \tag{16}$$

Resource Allocation:

$$\min_{x_t} gNN(\mathbf{z}_t, \mathbf{x}_t) \tag{17}$$

Reinforcement Learning Objective:

$$\max \textstyle\sum_{t=0}^{T} \gamma^t (f\mathrm{GNN}(\mathbf{z}_t, \mathbf{x}_t) - \gamma g\mathrm{GNN}\,(\mathbf{z}_t, \mathbf{c}_t)) \tag{18}$$

By integrating advanced GNN techniques with RL algorithms, we develop robust and scalable models for dynamic yield optimization and resource allocation. This leads to significant improvements in efficiency, productivity, and cost reduction in industrial systems.

3. Results and Discussion

Let's consider a simple numerical example to illustrate the concepts of dynamic yield optimization and resource allocation using Graph Neural Networks (GNNs) and Reinforcement Learning (RL) in an industrial system.

Example Setup
Industrial System:
We have a small manufacturing system with three machines ($M1, M2, M3$) that interact with each other. The interactions are represented as a dynamic graph $\mathbf{G_t} = (\mathbf{V_t}, \mathbf{E_t})$ at time $\boldsymbol{t}$.

- $\mathbf{V_t}$= $\{M1, M2, M3\}$
- $\mathbf{E_t} = \{(M1, M2), (M2, M3), (M3, M1)\}$

Each machine $M1$ has a feature vector $\mathbf{h_{Mi}^t}$ representing its state, such as operational efficiency, temperature, and workload.

Feature Vectors:

- $\mathbf{h_{M1}^t} =$ [0.8,70,0.5} (efficiency: 0.8, temperature: 70°C, workload: 0.5)
- $\mathbf{h_{M2}^t} =$ [0.7,65,0.6]
- $\mathbf{h_{M3}^t} =$ [0.9,75,0.4]

Node and Edge Embeddings

We learn node and edge embeddings through a GNN. Assume we have one GNN layer with the following parameters:

- Weight matrices: $\boldsymbol{w} = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \end{bmatrix}, \boldsymbol{w_0} = \begin{bmatrix} 0.2 & 0.3 & 0.4 \\ 0.5 & 0.6 & 0.7 \\ 0.8 & 0.9 & 1.0 \end{bmatrix}$
- Activation function: ReLU

The update rule for node $M1$ is:

$$\mathbf{h}_{\boldsymbol{M1}}^{(1),t} = \sigma\left(\alpha_{M1.M2}^{t}\, \mathbf{W}\mathbf{h}_{M2}^{t} + \alpha_{M1,M3}^{t}\mathbf{W}\mathbf{h}_{M3}^{t} + \mathbf{W}_0\mathbf{h}_{M1}^{t}\right)$$

Assume attention coefficients $\alpha_{M1.M2}^{t} = \alpha_{M1,M3}^{t} = 0.5$

Calculating embeddings for $M1$:

$$\mathbf{h}_{\boldsymbol{M1}}^{(1),t} = \sigma\left(0.5\begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \end{bmatrix}\begin{bmatrix} 0.7 \\ 65 \\ 0.6 \end{bmatrix} 0.5 + \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \end{bmatrix}\begin{bmatrix} 0.9 \\ 75 \\ 0.4 \end{bmatrix} + \begin{bmatrix} 0.2 & 0.3 & 0.4 \\ 0.5 & 0.6 & 0.7 \\ 0.8 & 0.9 & 1.0 \end{bmatrix}\begin{bmatrix} 0.8 \\ 70 \\ 0.5 \end{bmatrix}\right)$$

Let's break down the matrix multiplications:

$$\mathbf{h}_{\boldsymbol{M2}}^{\boldsymbol{t}} = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \end{bmatrix}\begin{bmatrix} 0.7 \\ 65 \\ 0.6 \end{bmatrix} = \begin{bmatrix} 13.09 \\ 34.91 \\ 56.73 \end{bmatrix}$$

$$\mathbf{h}_{\boldsymbol{M2}}^{\boldsymbol{t}} = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \end{bmatrix}\begin{bmatrix} 0.9 \\ 75 \\ 0.4 \end{bmatrix} = \begin{bmatrix} 15.23 \\ 40.6 \\ 65.97 \end{bmatrix}$$

$$\mathbf{W}_0\mathbf{h}_{M1}^{t} = \begin{bmatrix} 0.2 & 0.3 & 0.4 \\ 0.5 & 0.6 & 0.7 \\ 0.8 & 0.9 & 1.0 \end{bmatrix}\begin{bmatrix} 0.8 \\ 70 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 21.5 \\ 43.8 \\ 66.1 \end{bmatrix}$$

Combining and applying the ReLU activation function:

$$\mathbf{h}_{\boldsymbol{M1}}^{(1),t} = \sigma\left(0.5\begin{bmatrix} 13.09 \\ 34.91 \\ 56.73 \end{bmatrix} 0.5 + \begin{bmatrix} 15.23 \\ 40.6 \\ 65.97 \end{bmatrix} + \begin{bmatrix} 21.5 \\ 43.8 \\ 66.1 \end{bmatrix}\right)$$

$$\mathbf{h}_{\boldsymbol{M1}}^{(1),t} = \sigma\left(\begin{bmatrix} 16.545 \\ 17.455 \\ 28.365 \end{bmatrix} + \begin{bmatrix} 7.615 \\ 20.3 \\ 32.985 \end{bmatrix} + \begin{bmatrix} 21.5 \\ 43.8 \\ 66.1 \end{bmatrix}\right)$$

$$\mathbf{h}_{\boldsymbol{M1}}^{(1),t} = \sigma\left(\begin{bmatrix} 35.66 \\ 81.55 \\ 127.45 \end{bmatrix}\right)$$

$$\mathbf{h}_{\boldsymbol{M1}}^{(1),t} = \begin{bmatrix} 35.66 \\ 81.55 \\ 127.45 \end{bmatrix}$$

Dynamic Yield Optimization

Given the updated node embeddings, the GNN predicts the yield $\boldsymbol{y_t}$.
Assume the yield function is a linear combination of the node embeddings:

$$\boldsymbol{y_t} = fGNN\ (\boldsymbol{z_t}, \boldsymbol{x_t}) = \sum_{v \in \boldsymbol{V_t}} \boldsymbol{w_v}.\, \mathbf{h}_{\boldsymbol{v}}^{\boldsymbol{t}}$$

Where $\boldsymbol{w_v}$ are weights for each machine's contribution to the yield.
Assume:
$\boldsymbol{w_{M1}} =$ [0.5,0.3,0.2]
$\boldsymbol{w_{M2}} =$ [0.4,0.4,0.2]

$\boldsymbol{w_{M3}} = [0.6,0.2,0.2]$

The predicted yield is:

$\boldsymbol{y_t} = \boldsymbol{w_{M1}}.\mathbf{h}_{\boldsymbol{M1}}^{(1),\boldsymbol{t}} + \boldsymbol{w_{M2}} + \mathbf{h}_{\boldsymbol{M2}}^{(1),\boldsymbol{t}} + \boldsymbol{w_{M3}} + \mathbf{h}_{\boldsymbol{M3}}^{(1),\boldsymbol{t}}$

$\boldsymbol{y_t} = [0.5,0.3,0.2] \cdot [35.66,81.555,127.45] + [0.4,0.4,0.2] \cdot [0.7,65,0.6] + [0.6,0.2,0.2] \cdot [0.9,75,0.4]$

$\boldsymbol{y_t} = 0.5 \cdot 35.66 + 0.3 \cdot 81.555 + 0.2 \cdot 127.45 + 0.4 \cdot 0.7 + 0.4 \cdot 65 + 0.2 \cdot 0.6 + 0.6 \cdot 0.9 + 0.2 \cdot 75 + 0.2 \cdot 0.4$

$\boldsymbol{y_t} = 17.83 + 24.4665 + 25.49 + 0.28 + 26 + 0.12 + 0.54 + 15 + 0.08$

$\boldsymbol{y_t} = 109.8065$

Resource Allocation

To optimize resource allocation $\boldsymbol{r_t}$, assume the cost function is also a linear combination of the node embeddings:

$\boldsymbol{c_t} = gGNN(\boldsymbol{z_t}, \boldsymbol{c_t}) = \sum_{v \in V_t} \mathbf{u}_v.\mathbf{h}_v^t$

where $\mathbf{u}_v$ are weights for each machine's contribution to the cost.

Assume:

$\mathbf{u}_{\boldsymbol{M1}} = [0.3,0.3,0.4]$

$\mathbf{u}_{\boldsymbol{M2}} = [0.2,0.5,0.3]$

$\mathbf{u}_{\boldsymbol{M3}} = [0.4,0.3,0.3]$

The predicted cost is:

$\boldsymbol{c_t} = \boldsymbol{u_{M1}}.\mathbf{h}_{\boldsymbol{M1}}^{(1),\boldsymbol{t}} + \boldsymbol{u_{M2}} + \mathbf{h}_{\boldsymbol{M2}}^{(1),\boldsymbol{t}} + \boldsymbol{u_{M3}} + \mathbf{h}_{\boldsymbol{M3}}^{(1),\boldsymbol{t}}$

$\boldsymbol{c_t} = [0.3,0.3,0.4] \cdot [35.66,81.555,127.45] + [0.2,0.5,0.3] \cdot [0.7,65,0.6] + [0.4,0.3,0.3] \cdot [0.9,75,0.4]$

$\boldsymbol{c_t} = 0.3 \cdot 35.66 + 0.3 \cdot 81.555 + 0.4 \cdot 127.45 + 0.2 \cdot 0.7 + 0.5 \cdot 65 + 0.3 \cdot 0.6 + 0.4 \cdot 0.9 + 0.3 \cdot 75 + 0.3 \cdot 0.4$

$\boldsymbol{c_t} = 10.698 + 24.4665 + 50.98 + 0.14 + 32.5 + 0.18 + 0.36 + 22.5 + 0.12$

$\boldsymbol{c_t} = 141.9445$

Reinforcement Learning Integration

State Representation:

The state $\boldsymbol{s_t}$ is derived from the GNN embeddings $\boldsymbol{z_t}$, e. g. , $\boldsymbol{s_t} = \mathbf{h}_v^{(1),\boldsymbol{t}}$.

Action Selection:

Actions a_t could be adjusting parameters like x_t and resource allocations r_t.

Reward Function:

$\boldsymbol{R_t} = \boldsymbol{y_t} - \boldsymbol{\lambda c_t}$

Assume λ =0.01:

$\boldsymbol{R_t} = 109.8065 - 0.01 \cdot 141.9445 = 109.8065 - 1.419445 = 108.387055$

In this numerical example, we demonstrated the integration of Graph Neural Networks (GNNs) and Reinforcement Learning (RL) for dynamic yield optimization and resource allocation in a small manufacturing system with three machines ($M1, M2,$ and $M3$).

Node Embeddings Calculation:

The feature vectors for each machine were:

$\mathbf{h}_{\boldsymbol{M1}}^{(1),\boldsymbol{t}} = [0.8,70,0.5]$, $\mathbf{h}_{\boldsymbol{M2}}^{(1),\boldsymbol{t}} = [0.7,65,0.6]$, $\mathbf{h}_{\boldsymbol{M3}}^{(1),\boldsymbol{t}} = [0.7,65,0.6]$

Using a GNN layer with weight matrices and ReLU activation, we computed the updated embedding for $\boldsymbol{M1}$ as:

$\mathbf{h}_{\boldsymbol{M1}}^{(1),\boldsymbol{t}} = [35.66,81.555,127.45]$

Dynamic Yield Calculation:

The yield function, defined as a linear combination of the node embeddings with weights w_v, resulted in a predicted yield of:

$\boldsymbol{y_t} = 109.8065$

Resource Allocation and Cost Calculation:
The cost function, defined similarly as a linear combination of node embeddings with weights $\mathbf{u}_v$, resulted in a predicted cost of:
$\boldsymbol{y_t} = 141.9445$
Reinforcement Learning Reward Calculation:
The reward function, considering both yield and cost, was calculated as:
$\boldsymbol{R_t} = \boldsymbol{y_t} - \boldsymbol{\lambda c_t} = 108.387055$ (with $\lambda = 0.01$)

Discussion
This example highlights several important aspects of using GNNs and RL for industrial optimization:
Efficacy of GNNs:

1) GNNs effectively captured the complex interactions between machines in the manufacturing system, providing meaningful embeddings that reflect the machines' states and interactions.
2) The use of attention coefficients in the GNN layer allowed for weighted aggregation of neighboring nodes' information, enhancing the model's ability to focus on relevant interactions.

Yield and Cost Optimization:

1) The yield and cost functions, as linear combinations of node embeddings, provided a straightforward yet powerful method for predicting these key metrics.
2) The calculated yield ($\boldsymbol{y_t} = 109.8065$) and cost ($\boldsymbol{c_t} = \mathbf{141.9445}$) demonstrate the model's capability to evaluate the system's performance based on current states**.**

Reinforcement Learning for Dynamic Optimization:

1) The RL framework's reward function incorporated both yield and cost, guiding the agent to balance these factors for overall system optimization.
2) The reward ($\boldsymbol{R_t}$=108.387055) reflects the trade-off between maximizing yield and minimizing cost, which is crucial for efficient resource allocation in industrial operations.

Scalability and Real-World Application:

1) While this example used a small-scale system, the approach can be scaled to larger, more complex industrial settings with many machines and interactions.
2) The integration of GNNs and RL can adapt to dynamic changes in the system, making it suitable for real-time decision-making and continuous improvement in industrial processes.

This numerical example demonstrates the potential of combining GNNs and RL for dynamic yield optimization and resource allocation in industrial systems. The results indicate that this approach can effectively capture the complexities of machine interactions and optimize key performance metrics, paving the way for more efficient and adaptive industrial operations.

4. Conclusion

This research demonstrates the powerful synergy between Graph Neural Networks (GNNs) and Reinforcement Learning (RL) for dynamic yield optimization and resource allocation in industrial systems. Through a numerical example, we showcased how GNNs can effectively capture and model the complex interactions between different machines within a manufacturing setup. The incorporation of attention mechanisms allowed the model to prioritize significant interactions, enhancing its predictive power. GNNs successfully captured the nuanced interactions between machines, leading to meaningful embeddings that accurately represent the state and performance of each machine. By leveraging GNN-derived embeddings, we were able to predict both yield and cost through linear combination functions. This approach provided a clear and effective method for evaluating key performance metrics, essential for decision-making in industrial operations. The integration of RL facilitated the dynamic optimization of yield and cost, where the reward function balanced these two critical factors. This enabled the system to adaptively allocate resources and adjust operational parameters in real time, aiming for overall system optimization. Although the example involved a small-scale system, the methodology is scalable to more complex industrial environments. The combination of GNNs and RL is well-suited for real-world applications, offering a robust framework for continuous improvement and adaptive decision-making

in industrial processes. The successful implementation of this approach in a numerical example underscores its potential for broader application in various industrial contexts. Future work could explore more sophisticated models, larger-scale implementations, and integration with other advanced optimization techniques to further enhance performance and adaptability. The integration of GNNs and RL represents a promising frontier for optimizing industrial systems. This research paves the way for more efficient, adaptive, and intelligent industrial operations, ultimately contributing to significant improvements in yield, cost-efficiency, and overall system performance.

References

[1] C. Li, Y. Chen, and Y. Shang, "A review of industrial big data for decision making in intelligent manufacturing," *Eng. Sci. Technol. an Int. J.*, vol. 29, no. 23, p. 101021, 2022, doi: https://doi.org/10.1016/j.jestch.2021.06.001.

[2] T. Ahmad, R. Madonski, D. Zhang, C. Huang, and A. Mujeeb, "Data-driven probabilistic machine learning in sustainable smart energy/smart energy systems: Key developments, challenges, and future research opportunities in the context of smart grid paradigm," *Renew. Sustain. Energy Rev.*, vol. 160, no. 34, p. 112128, 2022, doi: https://doi.org/10.1016/j.rser.2022.112128.

[3] T. Cerquitelli *et al.*, "Manufacturing as a data-driven practice: methodologies, technologies, and tools," *Proc. IEEE*, vol. 109, no. 4, pp. 399–422, 2021, doi: https://doi.org/10.1109/JPROC.2021.3056006.

[4] L. Wu, W. Ji, B. Feng, U. Hermann, and S. AbouRizk, "Intelligent data-driven approach for enhancing preliminary resource planning in industrial construction," *Autom. Constr.*, vol. 130, no. 16, p. 103846, 2021, doi: https://doi.org/10.1016/j.autcon.2021.103846.

[5] M. H. Ali and M. S. U. Talukder, "Increasing water productivity in crop production—A synthesis," *Agric. water Manag.*, vol. 95, no. 11, pp. 1201–1213, 2008, doi: https://doi.org/10.1016/j.agwat.2008.06.008.

[6] M. A. Berry and D. A. Rondinelli, "Proactive corporate environmental management: A new industrial revolution," *Acad. Manag. Perspect.*, vol. 12, no. 2, pp. 38–50, 1998, doi: https://doi.org/10.5465/ame.1998.650515.

[7] B. Khemani, S. Patil, K. Kotecha, and S. Tanwar, "A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions," *J. Big Data*, vol. 11, no. 1, p. 18, 2024, doi: https://doi.org/10.1186/s40537-023-00876-4.

[8] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. neural networks Learn. Syst.*, vol. 32, no. 1, pp. 4–24, 2020, doi: https://doi.org/10.1109/TNNLS.2020.2978386.

[9] J. Zhang, Y. Liu, Z. Li, and Y. Lu, "Forecast-assisted service function chain dynamic deployment for SDN/NFV-enabled cloud management systems," *IEEE Syst. J.*, vol. 17, no. 3, pp. 4371–4382, 2023, doi: https://doi.org/10.1109/JSYST.2023.3263865.

[10] F. Cui, Z. Jiang, X. Zhou, J. Zheng, and N. Geng, "A configuration optimization approach for reconfigurable manufacturing system based on column-generation combined with graph neural network," *Int. J. Prod. Res.*, pp. 1–22, 2024, doi: https://doi.org/10.1080/00207543.2024.2366992.

[11] T.-D. Nguyen, T. Le-Cong, T. H. Nguyen, X.-B. D. Le, and Q.-T. Huynh, "Toward the analysis of graph neural networks," in *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: New Ideas and Emerging Results*, 2022, pp. 116–120. doi: https://doi.org/10.1145/3510455.3512780.

[12] L. Waikhom and R. Patgiri, "Graph neural networks: Methods, applications, and opportunities," in *arXiv preprint arXiv:2108.10733*, 2021, pp. 1–96. doi: https://doi.org/10.48550/arXiv.2108.10733.

[13] W. Fan, P. Lu, K. Pang, R. Jin, and W. Yu, "Linking entities across relations and graphs," *ACM Trans. Database Syst.*, vol. 49, no. 1, pp. 1–50, 2024, doi: https://doi.org/10.1145/3639363.

[14] A. Lysenko, I. A. Roznovăţ, M. Saqi, A. Mazein, C. J. Rawlings, and C. Auffray, "Representing and querying disease networks using graph databases," *BioData Min.*, vol. 9, no. 33, pp. 1–19, 2016, doi: https://doi.org/10.1186/s13040-016-0102-8.

[15] M. Liu, X. Li, J. Li, Y. Liu, B. Zhou, and J. Bao, "A knowledge graph-based data representation approach for IIoT-enabled cognitive manufacturing," *Adv. Eng. Informatics*, vol. 51, p. 101515, 2022, doi: https://doi.org/10.1016/j.aei.2021.101515.

[16] X. Li, M. Lyu, Z. Wang, C.-H. Chen, and P. Zheng, "Exploiting knowledge graphs in industrial products and services: a survey of key aspects, challenges, and future perspectives," *Comput. Ind.*, vol. 129, no. 21, p. 103449, 2021, doi: https://doi.org/10.1016/j.compind.2021.103449.

[17] V. A. Varma, G. V Reklaitis, G. E. Blau, and J. F. Pekny, "Enterprise-wide modeling & optimization—An overview of emerging research challenges and opportunities," *Comput. Chem. Eng.*, vol. 31, no. 5–6, pp. 692–711, 2007, doi: https://doi.org/10.1016/j.compchemeng.2006.11.007.

[18] Y. Teng, M. Liu, F. R. Yu, V. C. M. Leung, M. Song, and Y. Zhang, "Resource allocation for ultra-dense networks: A survey, some research issues and challenges," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, pp. 2134–2168, 2018, doi: https://doi.org/10.1109/COMST.2018.2867268.

[19] W. Boyd, W. S. Prudham, and R. A. Schurman, "Industrial dynamics and the problem of nature," *Soc. Nat. Resour.*, vol. 14, no. 7, pp. 555–570, 2001, doi: https://doi.org/10.1080/08941920120686.

[20] R. Isermann, "Process fault detection based on modeling and estimation methods—A survey," *automatica*, vol. 20, no. 4, pp. 387–404, 1984, doi: https://doi.org/10.1016/0005-1098(84)90098-0.

[21] H. Sarimveis, P. Patrinos, C. D. Tarantilis, and C. T. Kiranoudis, "Dynamic modeling and control of supply chain systems: A review," *Comput. Oper. Res.*, vol. 35, no. 11, pp. 3530–3561, 2008, doi: https://doi.org/10.1016/j.cor.2007.01.017.

[22] V. Oduguwa, A. Tiwari, and R. Roy, "Evolutionary computing in manufacturing industry: an overview of recent applications," *Appl. Soft Comput.*, vol. 5, no. 3, pp. 281–299, 2005, doi: https://doi.org/10.1016/j.asoc.2004.08.003.

[23] A. Hameed *et al.*, "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems," *Computing*, vol. 98, no. 3, pp. 751–774, 2016, doi: https://doi.org/10.1007/s00607-014-0407-8.

[24] X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang, "Graph neural networks and their current applications in bioinformatics," *Front. Genet.*, vol. 12, no. 2, p. 690049, 2021, doi: https://doi.org/10.3389/fgene.2021.690049.

[25] A. Gupta, P. Matta, and B. Pant, "Graph neural network: Current state of Art, challenges and applications," *Mater. Today Proc.*, vol. 46, no. 2, pp. 10927–10932, 2021, doi: https://doi.org/10.1016/j.matpr.2021.01.950.

[26] J. Skarding, B. Gabrys, and K. Musial, "Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey," *iEEE Access*, vol. 9, pp. 79143–79168, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3082932.

[27] A. Yaniv, P. Kumar, and Y. Beck, "Towards adoption of GNNs for power flow applications in distribution systems," *Electr. Power Syst. Res.*, vol. 216, no. 2, p. 109005, 2023, doi: https://doi.org/10.1016/j.epsr.2022.109005.

[28] L. Zhao, W. Jin, L. Akoglu, and N. Shah, "From stars to subgraphs: Uplifting any GNN with local structure awareness," in *arXiv preprint arXiv:2110.03753*, 2021, pp. 1–33. doi: https://doi.org/10.48550/arXiv.2110.03753.

[29] Y. Weng, X. Chen, L. Chen, and W. Liu, "GAIN: Graph attention & interaction network for inductive semi-supervised learning over large-scale graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 9, pp. 4257–4269, 2020, doi: https://doi.org/10.1109/TKDE.2020.3036212.

[30] W. Liao, B. Bak-Jensen, J. R. Pillai, Y. Wang, and Y. Wang, "A review of graph neural networks and their applications in power systems," *J. Mod. Power Syst. Clean Energy*, vol. 10, no. 2, pp. 345–360, 2021, doi: https://doi.org/10.35833/MPCE.2021.000058.

[31] Y. Li, C. Xue, F. Zargari, and Y. Li, "From Graph Theory to Graph Neural Networks (GNNs): The Opportunities of GNNs in Power Electronics," in *IEEE Access*, IEEE, 2023, pp. 145067–145084. doi: https://doi.org/10.1109/ACCESS.2023.3345795.

[32] J. F. Sallis, N. Owen, and M. J. Fotheringham, "Behavioral epidemiology: a systematic framework to classify phases of research on health promotion and disease prevention," *Ann. Behav. Med.*, vol. 22, no. 4, pp. 294–298, 2000, doi: https://doi.org/10.1007/BF02895665.

[33] J. Jalving, Y. Cao, and V. M. Zavala, "Graph-based modeling and simulation of complex systems," *Comput. Chem. Eng.*, vol. 125, no. 45, pp. 134–154, 2019, doi: https://doi.org/10.1016/j.compchemeng.2019.03.009.

[34] H. Kim, B. S. Lee, W.-Y. Shin, and S. Lim, "Graph anomaly detection with graph neural networks: Current status and challenges," *IEEE Access*, vol. 10, no. 10, pp. 111820–111829, 2022, doi: https://doi.org/10.1109/ACCESS.2022.3211306.

[35] B. Hancock, E. Ockleford, and K. Windridge, *An introduction to qualitative research*. Trent focus group London, 2001.

[36] D. R. Hancock, B. Algozzine, and J. H. Lim, *Doing case study research: A practical guide for beginning researchers*. Teachers College Press, 2021.

[37] I. van Gent and G. La Rocca, "Formulation and integration of MDAO systems for collaborative design: A graph-based methodological approach," *Aerosp. Sci. Technol.*, vol. 90, no. 7, pp. 410–433, 2019, doi: https://doi.org/10.1016/j.ast.2019.04.039.

[38] Z. Liu and J. Zhou, *Introduction to graph neural networks*. Springer Nature, 2022.

[39] J. Zhou *et al.*, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, no. 3, pp. 57–81, 2020, doi: https://doi.org/10.1016/j.aiopen.2021.01.001.

[40] A. Aamodt and M. Nygård, "Different roles and mutual dependencies of data, information, and knowledge—An AI perspective on their integration," *Data Knowl. Eng.*, vol. 16, no. 3, pp. 191–222, 1995, doi: https://doi.org/10.1016/0169-023X(95)00017-M.

[41] D.-E. Spanos, P. Stavrou, and N. Mitrou, "Bringing relational databases into the semantic web: A survey," *Semant. Web*, vol. 3, no. 2, pp. 169–209, 2012, doi: 10.3233/SW-2011-0055.