Journal of Process Control 123 (2023) 12–25

Contents lists available at ScienceDirect

Journal of Process Control

journal homepage: www.elsevier.com/locate/jprocont

Graph convolutional network soft sensor for process quality prediction

Mingwei Jia [a], Danya Xu [b], Tao Yang [b], Yi Liu [a,*], Yuan Yao [c,*]

[a] Institute of Process Equipment and Control Engineering, Zhejiang University of Technology, Hangzhou, 310023, People's Republic of China
[b] State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, People's Republic of China
[c] Department of Chemical Engineering, National Tsing Hua University, Hsinchu 30013, Taiwan

ARTICLE INFO

Article history:
Received 28 July 2022
Received in revised form 14 December 2022
Accepted 17 January 2023
Available online 24 January 2023

Keywords:
Soft sensor
Graph convolutional network
Quality prediction
Fermentation process

ABSTRACT

The nonlinear time-varying characteristics of the process industry can be modeled using numerous data-driven soft sensor methods. However, the intrinsic relationships among the variables, especially the localized spatial–temporal correlations that shed light on model behavior, have received little attention. In this study, a soft sensor based on a graph convolutional network is constructed by introducing the concept of graph to process modeling. The focus is on obtaining localized spatial–temporal correlations that aid in comprehending the intricate interactions among the variables included in the soft sensor. The model is trained by considering the regularization terms and it learns distinctive localized spatial–temporal correlations in an end-to-end manner. Furthermore, long-term dependence is established via temporal convolution. Thus, both the localized spatial–temporal correlations and time-series properties are captured. The feasibility of the proposed soft sensor is illustrated using two fermentation processes. The localized spatial–temporal correlations of this case study are visualized, and they demonstrate that the soft sensor is not a black-box model; instead, it is consistent with process knowledge.

1. Introduction

A highly accurate measurement of product quality is essential for monitoring, control, and optimization of modern industrial processes [1]. However, crucial quality indices/variables are not always measurable in real time, leading to a significantly delayed quality feedback. In such situations, soft sensors often play an important role in estimating the difficult-to-measure quality variables using easily accessible process knowledge or data [2]. First-principle models are preferred by some process engineers owing to their transparency. However, due to their complex process mechanisms, it is difficult to build these models quickly. [3]. On the other hand, data-driven models are gaining popularity, especially for complex processes where the available prior knowledge is either unavailable or insufficient [4–6].

In the past decade, data-driven models, including support vector regression (SVR) [7,8], various shallow neural network (NN) [9], partial least square (PLS) [10], and Bayesian network model (BNM) [11], have grown in popularity. Patanè and Xibilia adopted a recurrent NN to estimate key process variables in the sulfur recovery unit [12]. Desai et al. demonstrated that, in some cases, the SVR soft sensor is superior in comparison to the shallow NNs in terms of performance [13]. PLS-based soft sensors are popular in chemical engineering and chemometrics owing to their decent performance and simple design [14]. To improve the local prediction performance, these methods can be combined with the just-in-time learning techniques [15,16]. BNM can mine the causal relationship among variables and express it in the form of probability. Mohammadi et al. developed a soft sensor under the BNM framework for predicting sulfur content in a gas sweetening process [17]. However, in BNMs, parameter learning is a challenging task. In particular, both exact and approximate algorithms for calculating posteriors may lead to NP-hard problems, which are fatal for large models [11]. Additionally, with the increasing complexity of process industry, the structural limitations of these models make them insufficient for fully extracting representative features from historical data [18].

Recently, deep learning (DL) has received increasing attention in various fields including pattern recognition and non-destructive testing [19–21], and fault diagnosis [22]. DL exhibits powerful ability in historical data mining and helps describe complex processes [23,24]. Hence, DL has also been utilized for soft sensor development [25–28]. Xie et al. used long short-term memory (LSTM) to predict key variables and achieved satisfactory performance [29]. Huang et al. stacked autoencoder to identify significant variables and develop a soft sensor [30]. Chang et al. used contrastive learning to build a temporality-aware soft sensor that was robust to anomalies [31]. However, most current deep learning methods can be considered as "black boxes", which

* Corresponding authors.
E-mail addresses: yliuzju@zjut.edu.cn (Y. Liu), yyao@mx.nthu.edu.tw (Y. Yao).

https://doi.org/10.1016/j.jprocont.2023.01.010
0959-1524/

do not integrate prior knowledge and their behavior cannot be convincingly explained. Exploring variable relationships explicitly is beneficial for explaining model behavior and improving its interpretability.

Graphs can be extracted from various real-world relations among numerous entities and be defined using the vertex and edge sets, which represent the entities and their relationships, respectively [32]. Because graphs exploit essential and relevant relations among vertices, graph neural networks (GNNs) [33], which allow for the explicit study of variable relationships, have gained increasing popularity for capturing complex relationships. As a variant of traditional GNN, the graph convolutional network (GCN) [34] has exhibited powerful learning capability in several fields [35,36]. In the perspective of helping DL implement an explicit study of variable relationships, GCN have become noticeable as a representative model in the field of soft sensors.

Recent studies have introduced GCN into the field of soft sensors. Fang et al. used GCN to predict the quality of elements in the steelmaking process [37]. While the integration of relationships among elements in the form of prior knowledge improves model interpretability, the capturing of time-series properties alone is insufficient. Wang et al. established a soft sensor based on GCN and the gated recurrent unit for establishing relationships among variables and capturing their time-series properties during anaerobic digestion of kitchen waste [38]. As pointed by Song et al. it is indispensable to consider the localized spatial–temporal correlations and the cross-correlations of variables at different times [39]. Although the gated recurrent unit can capture cross-correlations to a certain extent, its capture process is unconstrained and not unique, which reduces model interpretability. Meanwhile, cross-correlation, as an adjunct of the gated recurrent unit has been rarely studied. The study of localized spatial–temporal correlations is meaningful for understanding the complex variable relationships in soft sensors.

In this study, we developed a novel GCN-based soft sensor utilizing localized spatial–temporal correlations. To reflect the relationship among variables at the same time and at different times, the proposed method uses spatial and temporal edges, respectively. The main contributions of this work are summarized as follows.

(a) We proposed a GCN-based soft sensor that aims to capture unique localized spatial–temporal correlations among variables by implementing the concept of graphs in the process industry.

(b) The proposed model autonomously learns the unique localized spatial–temporal graph that captures correlations. In addition, we used a regularization loss to constrain its learning process. Subsequently, the temporal convolution is developed to capture the time-series properties. Consequently, the GCN-based soft sensor facilitates a prediction that is consistent with the laws of physics and ensures model transparency.

(c) The performance of the GCN-based soft sensor is evaluated on two fermentation processes and is compared with that of several popular methods to prove its superiority. Contrary to the case of using a black-box model, the visualization of the localized spatial–temporal correlations using the proposed model exhibits its consistency with the prior knowledge.

2. Preliminaries

Definition 1. $G = (\mathbf{V}, \mathbf{A}, \mathbf{E})$ denotes the graph data, where $\mathbf{V}$ is the set of vertices (or nodes), $\mathbf{E}$ denotes the set of edges, and $\mathbf{A} \in \{0, 1\}^{V \times V}$ is the adjacency matrix of the graph G. G denotes the relationship of nodes in the spatial dimension, and the network structure does not change with time. In this work, the structure of G is that of a directed graph.

Definition 2. Graph node signal matrix of each sample is defined as $\mathbf{X}_G \in \mathbb{R}^{C \times V \times T}$, where C denotes the number of channels (i.e., the number of convolution kernels), V denotes the number of nodes, and T denotes the time step of each observed node. Additionally, each sample shares the same adjacency matrix $\mathbf{A}$.

2.1. Graph convolutional network

There are two key steps to set up a GCN. The first step is to perform the local information fusion on graph-structure data, and the second is to achieve graph representation learning, which embeds nodes or edges into vectors using a deep model. To perform these steps, hidden layer vectors are added in GCN through information transfer between adjacent nodes, and tasks such as classification and regression are completed through parameter learning. Fig. 1 shows the information transfer process of a two-layer GCN. For example, Node A in each hidden layer is derived from the information transfer of its neighboring nodes C, D, and itself (A). Then, the rectified linear unit (ReLU) activation function is exerted on Node A. Node A in Hidden Layer 2 is derived from hidden layer 1, and contains C, D, and itself.

GCN generalizes the traditional convolutional network from the Euclidean space to the vertex domain. This graph convolutional operation is based on Fourier transform and Laplacian matrix, and it can be formulated as follow:

$$\mathrm{GCN}(\mathbf{X}_G, \mathbf{A}) = \sigma\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{X}_G\mathbf{Q}\right) \tag{1}$$

where, $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ denotes the convolutional kernel; $\mathbf{D}$ is the degree matrix of the adjacency matrix $\mathbf{A}$; $\mathbf{X}_G$ is the input of the graph convolutional layer; $\mathbf{Q}$ is a weight matrix; σ denotes the activation function, such as ReLU.

2.2. Temporal convolutional network

Temporal convolutional networks (TCNs) [40] are based on two principles: (a) the network produces an output of the same length as the input; and (b) any leakage from the future into the past is impossible. First, the TCN is implemented as a regular one-dimensional convolution with a kernel of size $(1 \times a)$, where, a denotes the size of convolution kernel and each hidden layer has the same length as the input layer. Second, the TCN uses causal convolutions where output at the current time is convolved only with elements from the current time and earlier time. Fig. 2 shows the TCN framework with two hidden layers. Through the extraction of two hidden layers, the data at time t of the output layer captures the characteristics of the three previous moments and itself (t). TCN emphasizes building long and effective history sizes using a combination of deep networks and improving the ability of the networks to look as far as possible into the past to make a prediction.

3. Proposed soft sensor method

In this section, a GCN-based soft sensor method is proposed as follows. (1) The maximum information coefficients (MICs) [38] of each variable aimed at the target variable are calculated in the training set. (2) The validation set is used to determine the number of variables in the dataset. (3) The spatial–temporal convolutional layers (STCLs) are used to learn localized spatial–temporal correlations and to model long-term dependencies. (4) Multi-layer STCLs are stacked to form an encoding structure. (5) A fully connected layer (FCL) is added as a decoder to establish the mapping relationship between the data encoded by the stacked multi-layer STCLs and the target. It should be noted that the adjacency matrix of all layers is shared to ensure that the graph structures learned by the model are unique. The following sections detail the involved algorithms and model structures.

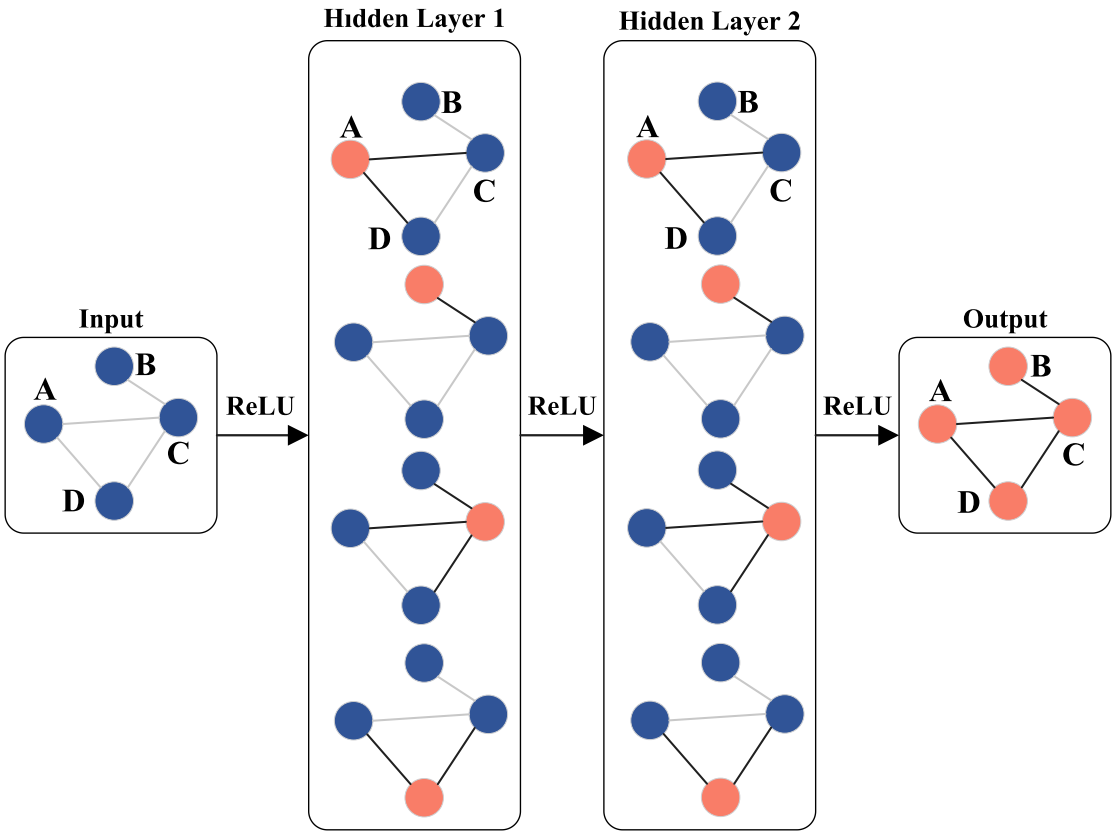


Fig. 1. A GCN structure with multiple graph convolutional layers.

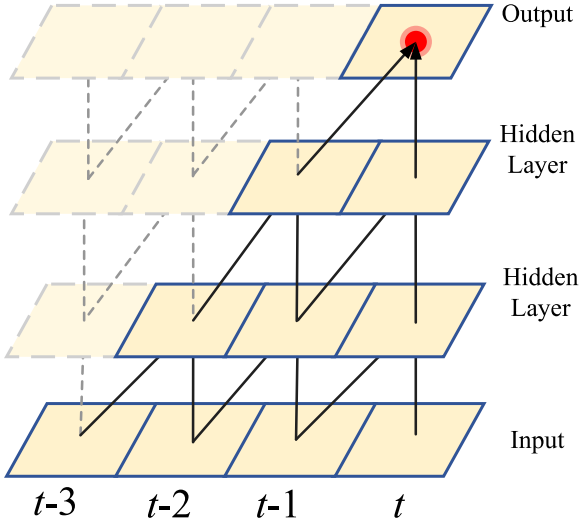


Fig. 2. A TCN framework with two hidden layers.

3.1. Variable selection based on MIC

An MIC [41] is derived from the mutual information (MI) that indicates the reduction in the uncertainty of a random variable caused by the introduction of another random variable. Here, only the MI between two discrete variables is considered. Given two discrete variables **j** and **u**, the joint probability distribution $p(\mathbf{j}, \mathbf{u})$ and MI can be obtained:

$$\mathrm{I}(\mathbf{j}, \mathbf{u}) = \sum_{j \in \mathbf{j}} \sum_{u \in \mathbf{u}} p(j, u) \log_2 \frac{p(j, u)}{p(j)\, p(u)}. \tag{2}$$

In addition, the MIC between **j** and **u** is the maximal normalized MI among all states:

$$\mathrm{MIC}_{j,u} = \max_{|j| \cdot |u| < B} \frac{\mathrm{I}(\mathbf{j}, \mathbf{u})}{\log_2 (\min(|j|, |u|))}, \tag{3}$$

where, $B = m^{0.55}$, and m denotes the size of the dataset. MIC gives each variable a full illustration of its importance relative to the key variable. After calculating the MIC between the variables and the target, an appropriate variable is selected as the model input.

3.2. Spatial–temporal convolutional layer

The STCL is divided into a localized spatial–temporal correlation module (LSCM) and a temporal convolution module (TCM). LSCM can directly capture the impact of each node on its neighbors that belong to both the current and the adjacent time steps. The most intuitive idea to achieve this goal is to connect all nodes with themselves at the adjacent time steps. As shown in Fig. 3(a), a localized spatial–temporal graph can be obtained by adding arrows between variables in two moments. For example, its effect of variable $\boldsymbol{x}_2$ on the remaining variables at the next moment follows a localized spatial–temporal correlation. According to the topological structure of the localized spatial–temporal graph, the correlations between each node and its spatial–temporal neighbors can be captured directly.

The relationship matrix ($\mathbf{A} \in \mathbb{R}^{V \times V}$) denotes the variable relationship matrix of the spatial graph, and the localized matrix ($\mathbf{A}' \in \mathbb{R}^{2V \times 2V}$) denotes the adjacency matrix of the localized spatial–temporal graph constructed on two continuous spatial graphs. Both these matrices are obtained through model training. As shown in Fig. 3(b), the $\mathbf{A}'$ is composed of three sub-matrices: $\mathbf{A}$, zero matrix $\mathbf{O}$, and time correlation matrix $\mathbf{A}_t$. $\mathbf{A}$ implies that

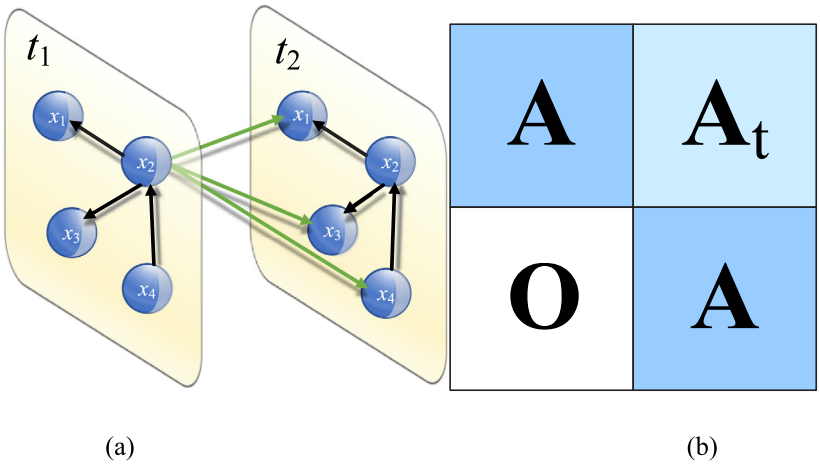


Fig. 3. (a) An example of a localized spatial–temporal graph, and (b) its adjacency matrix.

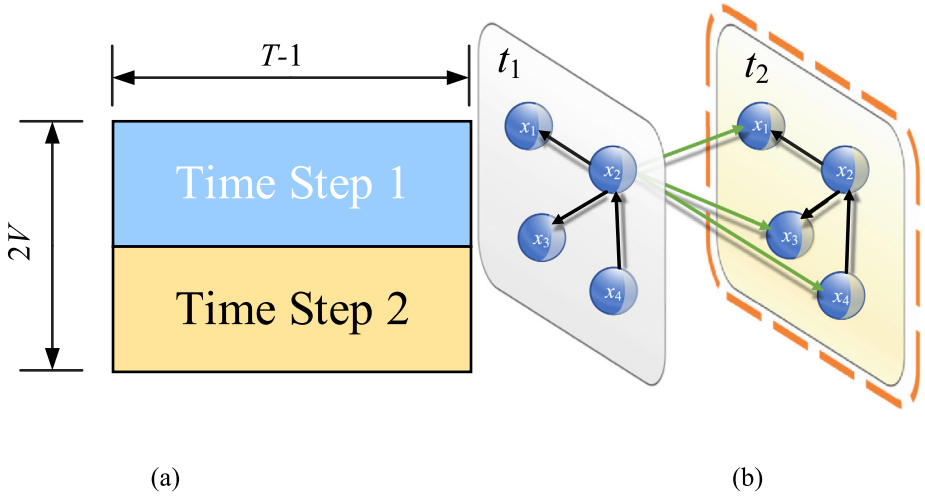


Fig. 4. For localized spatial–temporal graph: (a) the localized graph signal matrix, and (b) its cropping operation.

variables at different time steps share the same spatial structure, **O** implies that the previous time step is unaffected by the next time step, and $\mathbf{A}_t$ embodies the influence of the previous time step on the next time step. STCL built on a localized matrix can simultaneously capture the spatial and temporal relationships of variables.

The graph signal matrix $\mathbf{X}_G \in \mathbb{R}^{C\times V\times T}$ needs to be processed to correspond to $\mathbf{A}' \in \mathbb{R}^{2V\times 2V}$. As shown in Fig. 4(a), the graph signal matrix for each channel is $\{\mathbf{X}_1, \mathbf{X}_2,\ldots, \mathbf{X}_T\}$, $\mathbf{X}_i \in \mathbb{R}^{V\times 1}$, and it is transformed into $\{\{\mathbf{X}_1, \mathbf{X}_2,\ldots, \mathbf{X}_{T-1}\}, \{\mathbf{X}_2, \mathbf{X}_3,\ldots, \mathbf{X}_T\}\}$, $\mathbf{X}_i \in \mathbb{R}^{V\times 1}$. Finally, the localized graph signal matrix $\mathbf{X}'_G \in \mathbb{R}^{C\times 2V\times(T-1)}$ is obtained to correspond to $\mathbf{A}'$. The formula of LSCM can be expressed based on GCN as follows:

$$\mathrm{LSCM}\left(\mathbf{X}'_G, \mathbf{A}'\right) = \sigma\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}'\mathbf{D}^{-\frac{1}{2}}\mathbf{X}'_G\mathbf{Q}\right), \tag{4}$$

where $\mathbf{Q} \in \mathbb{R}^{(T-1)\times T}$ is a learnable matrix set for ensuring that the dimension remains unchanged after LSCM Subsequently, the cropping operation (Fig. 4(b)) removes all the features of the nodes in the previous time steps, and only the nodes in the next moment are retained. This is because LSCM has already aggregated the previous information. Each node contains the localized spatial–temporal correlations even after the previous time step is cropped.

The root mean square error (RMSE) is selected as the model loss. Because $\mathbf{A}'$ is obtained through independent training of the model, the loss is extended with a regularization term:

$$\begin{aligned} \mathrm{Loss} &= \sqrt{\tfrac{1}{T_m}\left(\mathbf{Y}-\mathbf{Y}'\right)^2} + \beta \mathrm{H}\left(\mathbf{A}'\right) \\ \mathrm{H}\left(\mathbf{A}'\right) &= -\mathbf{A}'\log 2\left(\mathbf{A}'\right) - \left(1-\mathbf{A}'\right)\log 2\left(1-\mathbf{A}'\right), \end{aligned} \tag{5}$$

where, **Y** and **Y'** denote the label and prediction, respectively. T_m is the number of time steps in a sample, β denotes a regularization coefficient, and H(.) indicates entropy. During training, the model may assign too much attention to irrelevant edges, resulting in the learning of wrong variable relationships. Therefore, a regularization term used to encourage discretization of the graph structure is added to the loss function, which improves the explainability of the learned graph structure.

TCM then encodes data that establish localized spatial–temporal correlations to capture temporal dynamic dependencies. An STCL is developed by combining LSCM and TCM. As presented in Fig. 5, all modules are followed by a batch normalization (BN) layer and an activation function.

3.3. Framework of GCN-based soft sensor

This section establishes the connection between the input variables and the target variable. Fig. 6 shows the GCN-based soft sensor framework and proposes a GCN-based modeling strategy. The proposed algorithm is summarized in Algorithm 1. After the

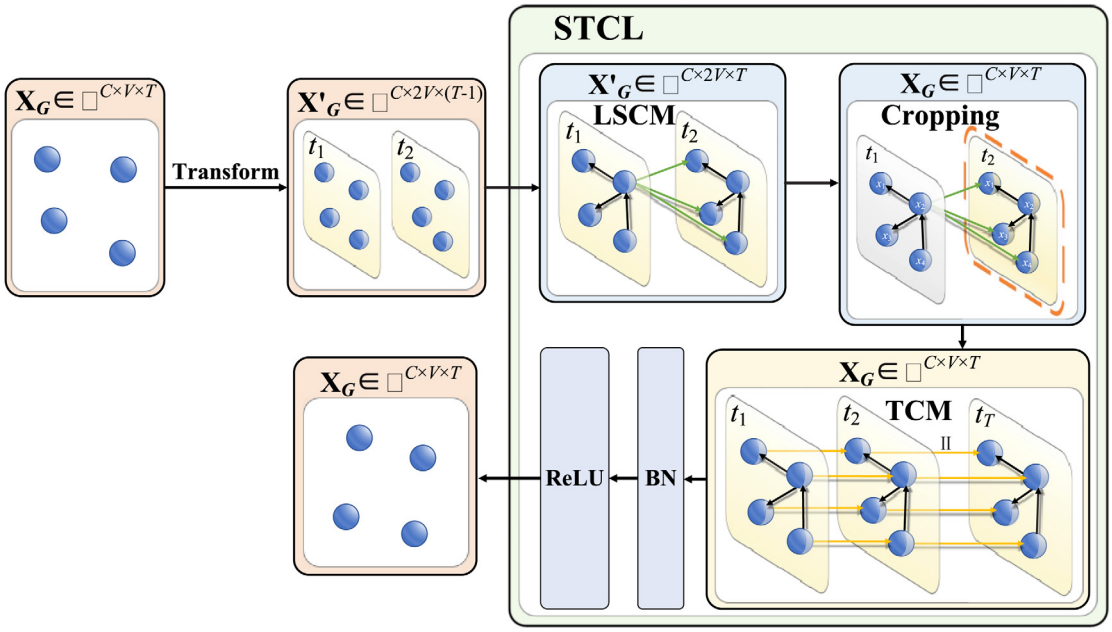


Fig. 5. The STCL framework.

Algorithm 1 The proposed algorithm.

Algorithm 1: The GCN-based soft sensor.

Input: The input variable $\mathbf{X}_G \in \mathbb{R}^{1\times V\times T}$, label $\mathbf{Y}$.

Output: The target variable $\mathbf{Y}'$.

Hyperparameters: Channel C, *epochs*, *learning rate*.

Initialize parameters the relationship matrix $\mathbf{A} \in \mathbb{R}^{V\times V}$, the time correlation matrix $\mathbf{A}_t \in \mathbb{R}^{V\times V}$, the zero matrix $\mathbf{O} \in \mathbb{R}^{V\times V}$, and the model parameters ($\mathbf{Q}$, $\boldsymbol{\theta}_{\mathrm{TCM}}$, $\boldsymbol{\theta}_{\mathrm{FCL}}$).

$\mathbf{A}' \in \mathbb{R}^{2V\times 2V} \leftarrow$ **concat(A, $\mathbf{A}_t$, O, A)**

$\mathbf{X}'_G \in \mathbb{R}^{1\times 2V\times (T-1)} \leftarrow$ transform($\mathbf{X}_G$)

for i =1, 2, …, *epoch* **do**

  $\mathbf{A}'[:, [0, V]]$ and $\mathbf{A}'[V{:}2V\text{-}1, V{:}2V\text{-}1] \leftarrow 0$

  $\mathbf{D} \in \mathbb{R}^{2V\times 2V} \leftarrow$ diagonal(sum($\mathbf{A}'$, axis=0))

  $\mathbf{X}'_G \in \mathbb{R}^{I\times V\times T} \leftarrow \sigma(\mathbf{D}^{-1/2}\mathbf{A}'\mathbf{D}^{-1/2}\mathbf{X}'_G\mathbf{Q})$

  $\mathbf{X}'_G \in \mathbb{R}^{C\times V\times T} \leftarrow \mathrm{TCM}_{\text{with channel } C}(\mathbf{X}'_G|\boldsymbol{\theta}_{\mathrm{TCM}})$

  $\mathbf{Y}' \leftarrow \mathrm{FCL}(\mathbf{X}'_G|\boldsymbol{\theta}_{\mathrm{FCL}})$

  $(\mathbf{Q}, \boldsymbol{\theta}_{\mathrm{TCM}}, \boldsymbol{\theta}_{\mathrm{FCL}}) \leftarrow (\mathbf{Q}, \boldsymbol{\theta}_{\mathrm{TCM}}, \boldsymbol{\theta}_{\mathrm{FCL}})$-*learning rate*$\times\nabla\mathrm{Loss}(\mathbf{Y}, \mathbf{Y}', \mathbf{A}')$

end

Return $\mathbf{Y}'$

local matrices are initialized, The STCL captures localized spatial–temporal correlations and time dynamic dependence. Because TCM involves convolution operations, multiple layers of STCLs are stacked to enlarge the receptive field of the model. In addition, the graph structure is shared by the multiple STCLs to ensure the uniqueness of the structure. The topological structure of the graph is captured by the model when the variable relationship is regarded as a graph. As a result of stacking multiple STCLs, each node contains the localized spatial–temporal correlations centered by itself.

4. Results and discussion

The production of secondary metabolites has been the subject of many studies due to its academic and industrial importance. Nevertheless, a primary obstacle to the implementation of control strategies is the lack of reliable sensors to measure the key variables, for example, biomass and production concentrations [1]. In this study, to verify the performance of the proposed method, we used a benchmark Pensim [42] and an industrial-scale Pensim (IndPensim) [43], both of which are simulation platforms for the penicillin fed-batch fermentation process. The code of this work is presented at https://github.com/MingweiJia/GCN-based_soft_sensor, including all code and Pensim/IndPensim datasets used.

4.1. Pensim

Typically, the penicillin fermentation process in Pensim has two operational phases, that is, batch and fed-batch. In general, the system switches to the fed-batch mode after approximately 44 h. Then, during the fed-batch operation, a constant feed is used. We observed that the bacteria did not secrete penicillin in the first stage. Therefore, we only established the soft sensor for

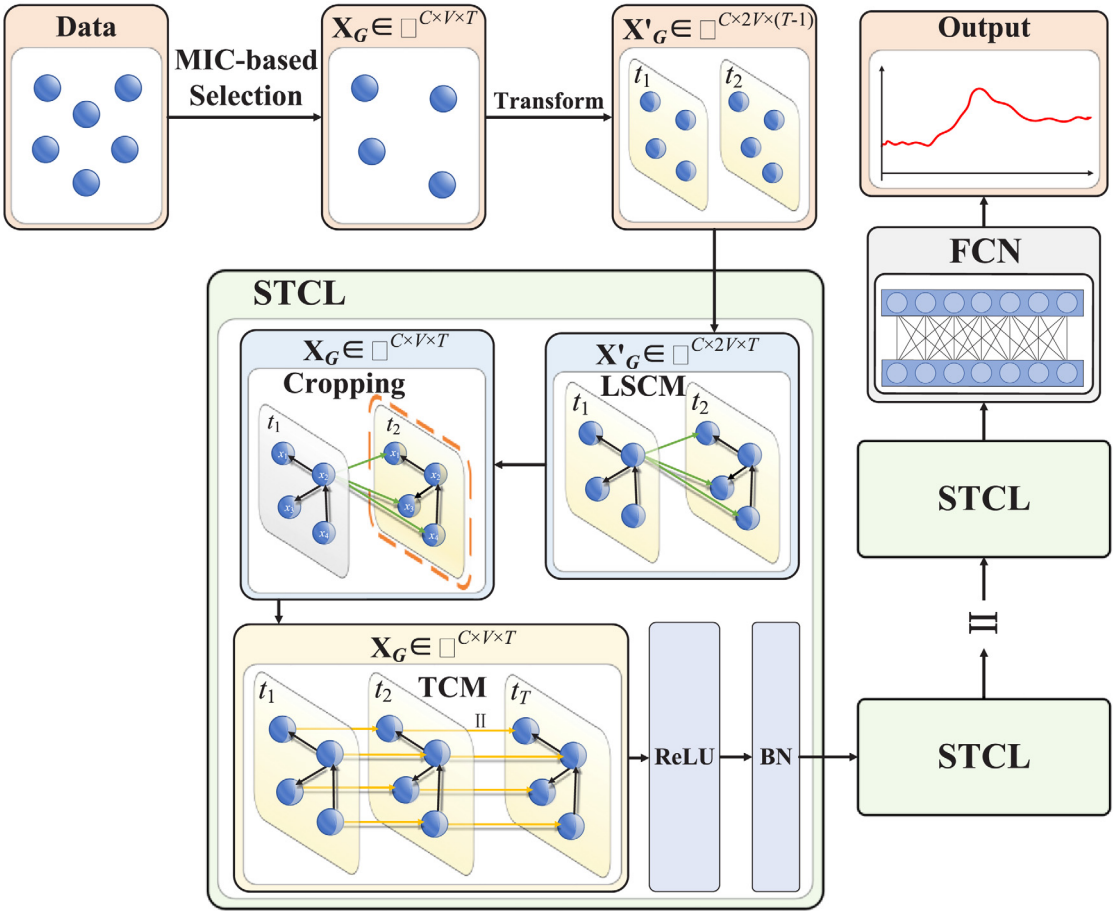


Fig. 6. The framework of GCN-based soft sensor model.

the second stage. The following 15 variables can be obtained from the penicillin fermentation process: aeration rate, agitator power, substrate feed flowrate, temperature, substrate concentration, dissolved oxygen concentration, biomass concentration, bacterial concentration, carbon dioxide concentration, pH value, generated temperature, and the flowrates of acid, base, cold water, and hot water. Among these, because the hot water flowrate is a constant value, and the bacterial concentration is an unmeasurable variable, these variables were omitted. Penicillin concentration (P) was considered as the target variable in this experiment.

The software Pensim [42] can be obtained from: http://www.chee.iit.edu/~control/software.html. It is used to generate an entire dataset containing 10 batches. The fermentation time of each batch was set to 400 h. In the actual fermentation process, to reduce the damage to the broth components, it is forbidden to frequently extract the broth to detect the product concentration. However, a relatively long detection interval would make it difficult to obtain dense labels required for model training in practice. Hence, to simulate the actual industrial production situation, the measurement frequency of penicillin concentration was set to once every 4 h, and that of the remaining variables was set to once every 1 h. We obtained one hundred pairs of data in every batch. After the first stage, 48 h were excluded, and data for 352 h per batch was obtained; 89 non-overlapping time windows were used to segment the 356 h data evenly in the time dimension. Finally, the dataset of dimension $10 \times 89 \times 4$ (*batches* $\times$ *samples* $\times$ *times*) is acquired. The generated data are divided into the training, validation, and test sets in a ratio of 4:3:3. We observed that disturbances and noise affected the training of the localized matrix. In particular, these may interfere with the data distribution, causing the model to learn a wrong relationship among variables, which decreases the reliability of the model. Although preprocessing, such as noise reduction, is necessary, this step was ignored in this specific simulation case. In the feature selection process, the training set was used to calculate the MIC between each process variable and the penicillin concentration P. The results thus obtained are listed in Table 1.

Table 1
MIC of each variable for penicillin concentration (P).

Variable	MIC
Aeration rate	0.9998
Agitator power	0.9997
Substrate feed flowrate	0.9999
Temperature	0.9780
Substrate concentration	0.9980
Dissolved oxygen concentration	0.9999
Biomass concentration	1
Carbon dioxide concentration	0.9999
pH value	0.9880
Generated temperature	1
Acid flowrate	0.1947
Base flowrate	0.9949
Cold water flowrate	0.8293

MIC is a tool of measuring information. A larger MIC value indicates a more significant linear or nonlinear correlation between two variables. In the feature selection phase of the proposed method, process variables are sorted according to the MIC values. Then, the SVR soft sensor model is built for quality prediction. The number of selected variables can be determined through model validation. As shown in Fig. 7, the validation loss calculated based

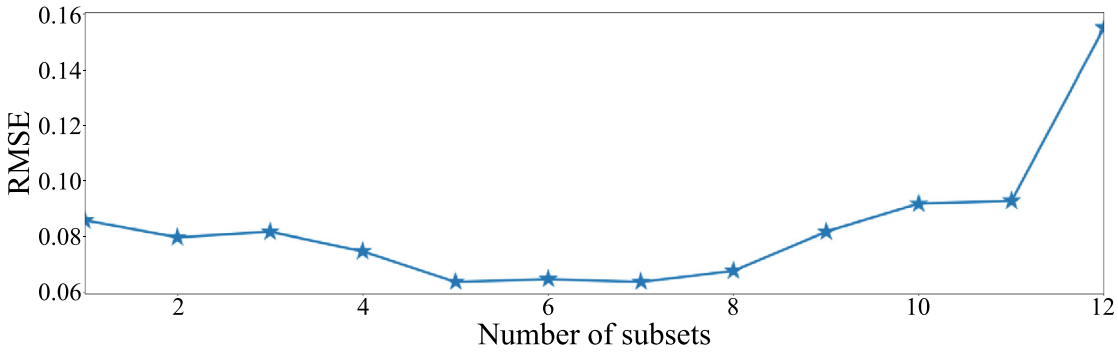


Fig. 7. Relationship between RMSE and the number of subsets.

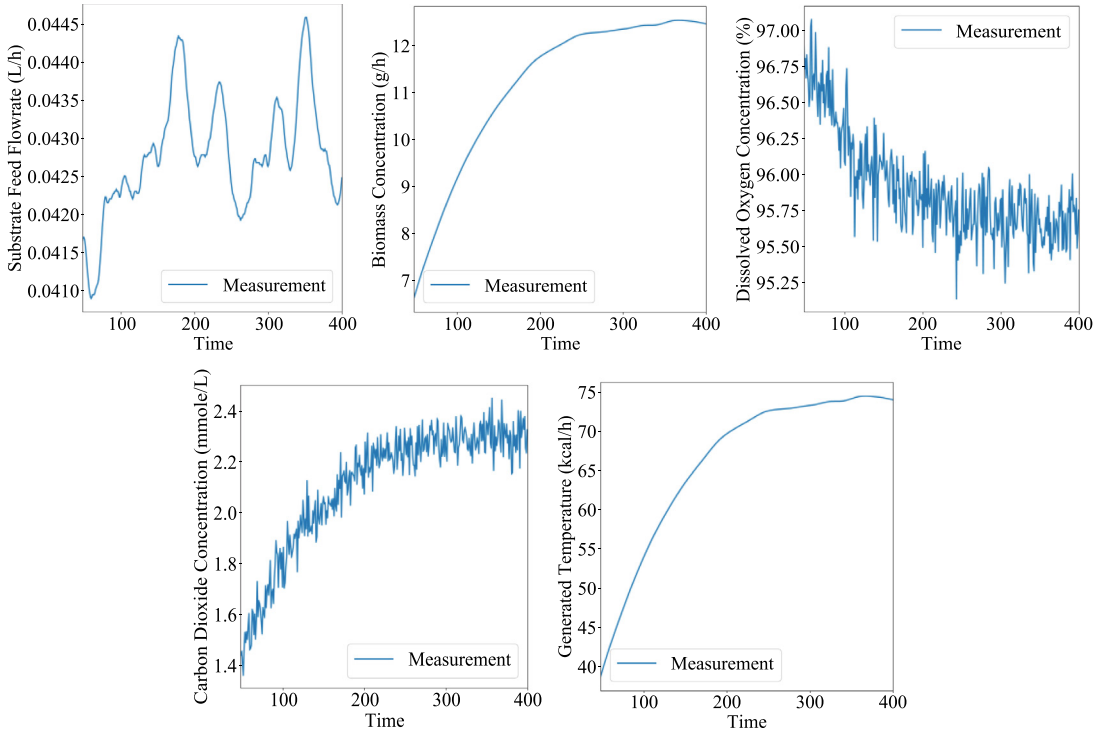


Fig. 8. Trajectories of input variables.

on RMSE reaches a minimum value when the number of selected variables is five, six, or seven. According to Occam's Razor, to maintain the simplicity of the model and reduce the number of model parameters, we selected the following five variables: substrate feed flowrate (V_{SF}), biomass concentration (E_B), dissolved oxygen concentration (E_{DO}), carbon dioxide concentration (E_{CO2}), and generated temperature (T_G). Here, V_{SF} is the manipulated variable. Plots of the input variables are shown in Fig. 8. Additionally, to describe the dynamic characteristics of the system more accurately, the value of P at the previous moment is added to the current soft sensor modeling. Therefore, P_{t-4} is added to the input as an autoregressive term of the previous moment for P_t.

The inputs and outputs of the model are as follows:

$$\begin{cases} \mathbf{x}_t = \begin{bmatrix} V_{SF_{t,t-1,t-2,t-3}}, E_{B_{t,t-1,t-2t,t-3}}, E_{DO_{t,t-1t,t-2,t-3}}, \\ E_{co_{2_{t,t-1,t-2,t-3}}}, T_{G_{t,t-1,t-2,t-3}}, P_{t-4} \end{bmatrix} \\ y_t = [P_t] \end{cases} \tag{6}$$

Because each sample contains data collected over the past 4 h, the kernel size of TCM is set to 1×3. Then, the validation set is used to test the performance of models with different combinations of channel numbers, where the channel number represents the number of convolution kernels in the TCM, and different channels contain information extracted by different convolution kernels. The final model will be determined from the three combinations of channel numbers (2, 4, 8), (4, 8, 16), and (8, 16, 32). RMSE of the three combinations of channel models is shown in Table 2. The model with channels (4, 8, 16) showed the best performance in the validation set. This is because the number of channels represents the size of the model capacity to a certain extent. Models with significantly small capacity are prone to underfitting, whereas those with considerably large capacity are prone to overfitting. Hence, (4, 8, 16) was selected as the number of model channels. We selected Adam-optimizer for gradient descent while setting its learning rate to 0.001. The loss function consists of two parts: prediction loss and entropy loss. β denotes a regularization coefficient of the entropy loss, which is used to constrain the sparsity of the learned localized matrix. During the model training process, a significantly large value of β results in an over-sparse matrix, whereas a considerably small value of β leads to an over-dense matrix. Both cases negatively affect the performance of the GNN-based model. In this work, the value of β in Eq. (5) is determined to be 0.08 according to the model performance on the validation set. Generally, the value on

Table 2
RMSE values of model with different channels.

Model	RMSE
GCN-based model with channels (2, 4, 8)	0.039
GCN-based model with channels (4, 8, 16)	0.020
GCN-based model with channels (8, 16, 32)	0.052

an edge connecting two uncorrelated variables is approximately zero. Therefore, a small number close to zero was assigned. Such edges should be removed according to a threshold when visualizing the adjacency matrix. In this work, this threshold is specified based on the 3-σ principle. In detail, the values smaller than the lower 3-σ limit are set to zero. Additionally, 5% and 10% Gaussian noise is added to the data to analyze the robustness of the model. All experiments were repeated five times, and the calculated average performance was considered as the final value. Four time-steps of historical data were used to predict the critical product quality of the present single time-step. The experimental environment consists of i7-9750H 2.6 Hz × 12 (CPU), RTX3090 24 GiB (GPU), 16 GB × 2 memory (DDR4), and Linux (OS).

We chose the following baseline methods for comparison: the common SVR [8], PLS [10], LSTM [29], and GCN [34]. Among these, PLS and SVM are popular, and LSTM is a commonly used time series model in DL. GCNs are known for their ability to capture variable correlations and were used in this study to verify the effectiveness of localized spatial–temporal correlations. Both PLS and SVM use grid search to select model parameters. The number of layers of LSTM and GCN was set to three, and the embedding dimension of each layer is (16, 32, 64) and the rest of the settings are consistent with the GCN-based model. The RMSE values of the baseline models and the GCN-based model are listed in Table 3. Compared with the SVR and PLS, the RMSE of the GCN-based model in Test 1 of free noise is lower 67.2% and 71.3%, respectively. This is mainly because SVM and PLS cannot effectively capture the spatial and temporal characteristics of the data. To show the complete fermentation process, the first stage, which is not modeled, is also added to the display in Fig. 9. Additionally, models emphasizing the importance of time, such as LSTM, showed good prediction performance. However, LSTM only considers its temporal characteristics and ignores spatial characteristics, resulting in an inferior prediction performance than the GCN-based model.

Furthermore, the GCN-based model outperforms others when the data contain 5% and 10% noise, which mainly relies on graph sparsification [44] and the low-pass filtering of GCN [45]. The former allows the model to capture more essential variable relationships by removing task-independent edges in the graph, which improves its generalization [44]. The latter allows the model to retain low-frequency and ignore high-frequency signals during multiple nodal signal propagations [45]. Because high-frequency signals correspond to noise in this study, the model exhibits improved noise reduction performance. It should be noted that the anti-noise performance discussed in this study is stabilization robustness, which is an adjunct capability of the model rather than a specific capability designed for noise. Therefore, the proposed model has good anti-noise performance in the process of learning localized spatial–temporal correlation, which further improves the prediction accuracy and reliability of the model.

The localized spatial–temporal correlation information learned by the model is visualized to prove its superior performance. The initial and output spatial topology structures of the relationship matrix are illustrated in Fig. 10, where V_{SF} is the operation variable and the other four are the process variables. The arrows represent the influence relationships among variables (including positive and negative correlations). Thus, by directly impacting E_B, the operating variable V_{SF} affects the remaining three process variables. Fig. 10(b) is explained as follows in combination with the fermentation mechanism [42]:

(a) The nutrient concentration per unit area is increased as the substrate feed flowrate V_{SF} increases. The biomass concentration E_B increases when a more favorable growth environment is received by bacteria. Therefore, an increase in V_{SF} is accompanied by an increase in E_B.

(b) Due to bacterial respiration, the dissolved oxygen concentration E_{DO} decreases and the CO_2 concentration increases. Simultaneously, the generated temperature T_G increases as a result of the heat generated by respiration.

(c) With increasing temperature, the solubility of oxygen in a solvent diminishes. As a result, increasing T_G results in a decrease in E_{DO}.

(d) An increase in dissolved oxygen concentration and decrease in CO_2 concentration and temperature allow bacteria to multiply rapidly. As a result, an increase in E_{DO}, and decrease in E_{CO2} and T_G contribute to an increase in E_B.

The topology structures of the localized matrix are illustrated in Fig. 11. Similar to the conclusion obtained from the relation matrix, in the localized spatial–temporal correlations (a) V_{SF} only affects E_B and (b) E_B affects the remaining variables.

Table 3
The test RMSE values of baseline models and GCN-based model.

Model	Free noise in test 1	5% Noise in test 1	10% Noise in test 1
SVR [8]	0.055	0.097	0.179
PLS [10]	0.063	0.098	0.186
LSTM [29]	0.032	0.076	0.134
GCN [34]	0.033	0.072	0.127
Proposed GCN	0.018	0.051	0.099
Model	Free noise in test 2	5% Noise in test 2	10% Noise in test 2
SVR [8]	0.046	0.082	0.167
PLS [10]	0.075	0.095	0.174
LSTM [29]	0.037	0.069	0.144
GCN [34]	0.038	0.076	0.137
Proposed GCN	0.012	0.053	0.093
Model	Free noise in test 3	5% Noise in test 3	10% Noise in test 3
SVR [8]	0.052	0.089	0.169
PLS [10]	0.063	0.091	0.172
LSTM [29]	0.035	0.061	0.135
GCN [34]	0.032	0.062	0.133
Proposed GCN	0.010	0.049	0.087

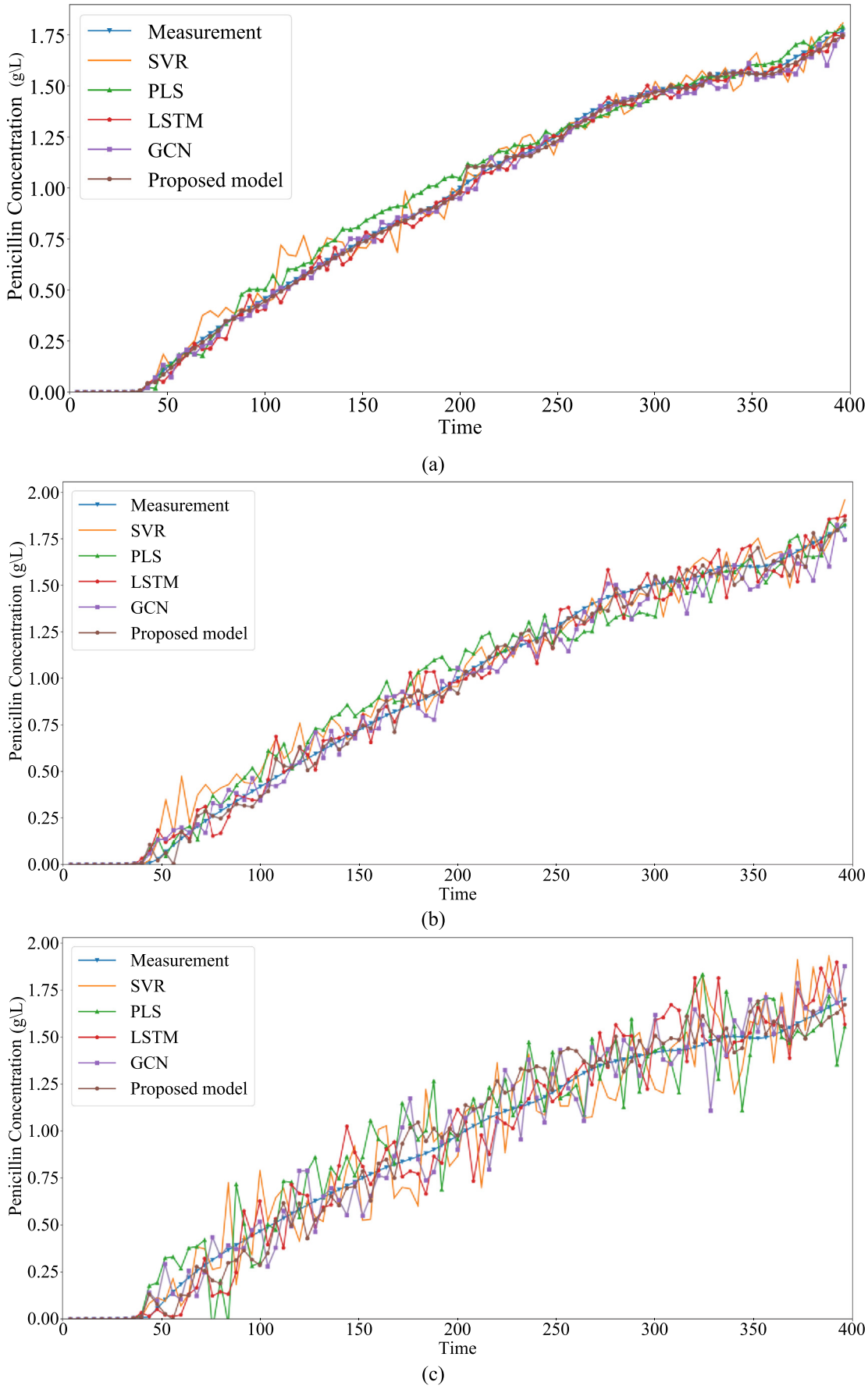


Fig. 9. Prediction in different scenarios: (a) free noise. (b) 5% noise, and (c) 10% noise.

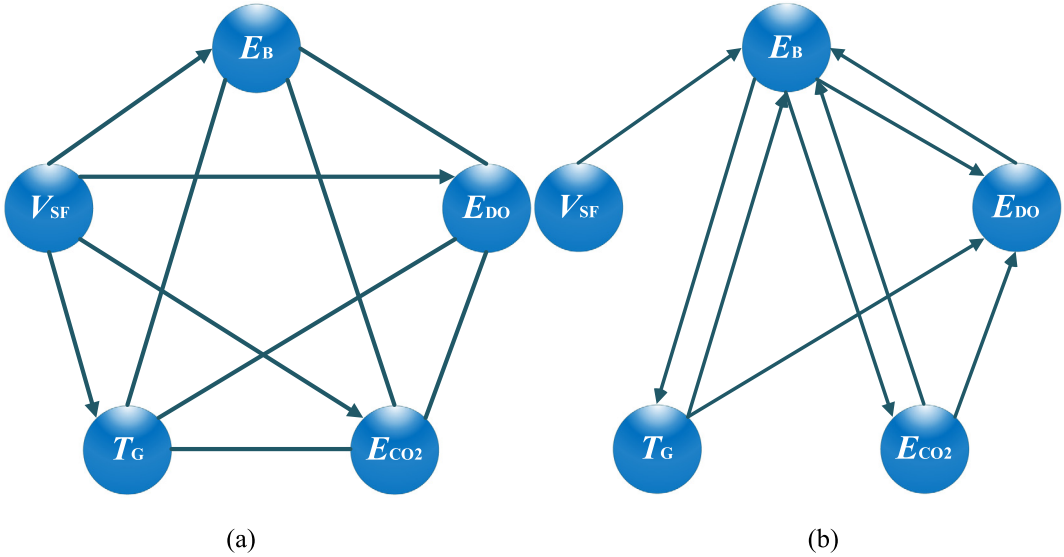


Fig. 10. (a) The initial topology structure of data, and (b) its output topology structure.

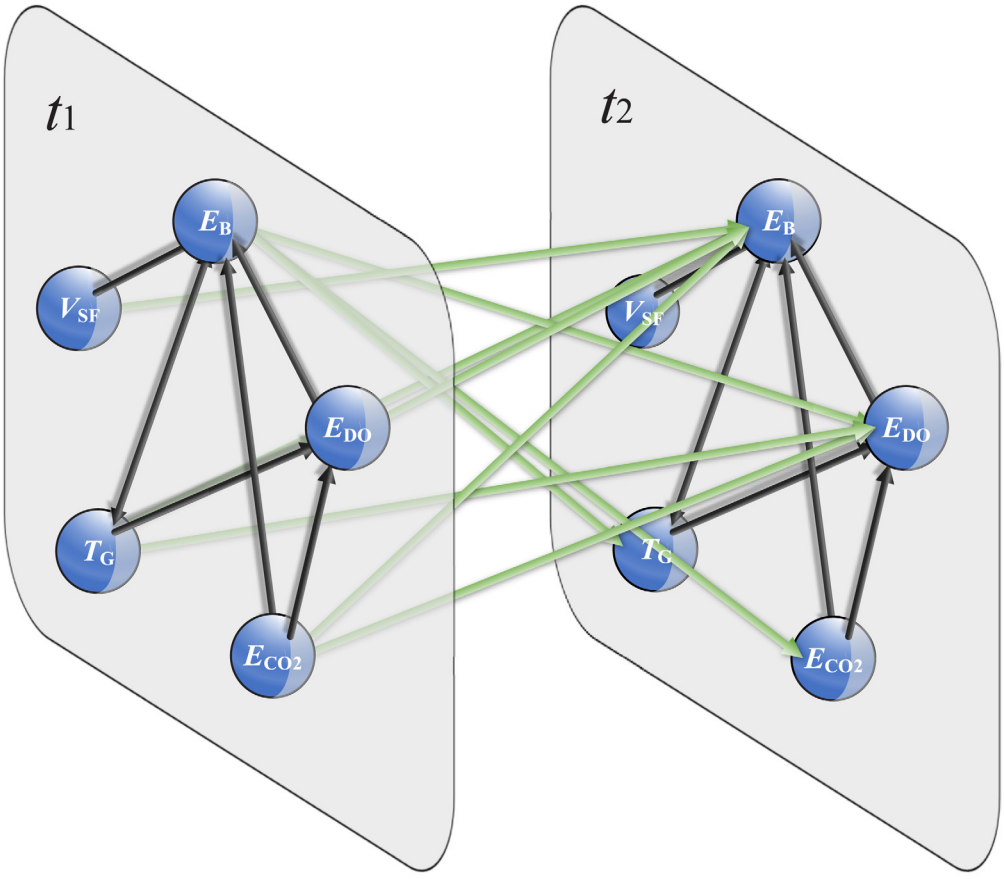


Fig. 11. The localized spatial–temporal correlation.

4.2. IndPensim

IndPensim is a simulator of industrial fed-batch fermentation processes (the dataset can be obtained from: www.industrialpenicillinsimulation.com) [43]. Compared with Pensim, soft sensor modeling in this case is more difficult because it is closer to the actual industrial process. The fermentation duration is between 168 h and 232 h, and the sensor records data every 0.2 h [43]. We used four normal batches (226 h, 230 h, 229 h, and 232 h) in this case. Similar to the Pensim experiment setup, the penicillin concentration measurement frequency is set to once every 1 h, and that for the remaining variables is set to once every 0.2 h. In IndPensim, the first phase lasts 24 h. Therefore, only the data of 24–226 h in each batch is selected to ensure the consistency of the dataset. Finally, the dataset of dimension 4 × 202 × 5 (*batches* × *samples* × *times*) is obtained. As opposed to the

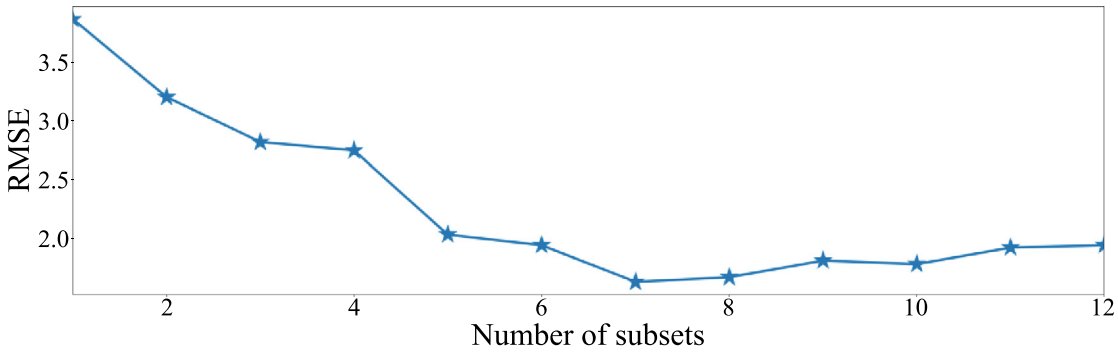


Fig. 12. Relationship between RMSE and number of subsets.

Table 4
MIC of each variable for penicillin concentration (P).

Variable	MIC
Basal flow rate	0.7998
Heating/cooling water flowrate	0.9997
Dissolved oxygen concentration	0.9282
Vessel volume	0.9980
Vessel weight	0.9980
pH value	0.1923
Temperature	0.9014
Generated temperature	0.8999
Carbon dioxide concentration in the off-gas	0.9980
Oxygen concentration in the off-gas	1
Carbon evolution rate	0.9947
Oxygen uptake rate	0.9949

Pensim experimental setup, only two batches were used as the training set, which further demonstrates the effectiveness of the GCN-based model in the scenario of limited training samples. Twelve variables in IndPensim were considered after removing the constant and step variables. The MIC between each process variable and the penicillin concentration P is calculated using the training set shown in Table 4.

Similarly, SVR soft sensor model was built for quality prediction. As shown in Fig. 12. the validation loss calculated based on RMSE reaches a minimum value when the number of selected variables is seven. According to Occam's Razor, to maintain the simplicity of the model and reduce the number of model parameters, we selected the following seven variables: heating/cooling water flow rate (R_W), vessel volume (V_V), vessel weight (V_W) carbon dioxide concentration in the off-gas (E_{CO2}), oxygen concentration in the off-gas (E_{O2}), carbon evolution rate (R_{CO2}), and oxygen uptake rate (R_{O2}). Here, R_W is the manipulated variable. P_{t-5} is added to the input as an autoregressive term of the previous moment for P_t. The inputs and outputs of the model are as follows:

$$\begin{cases} \mathbf{x}_t = \begin{bmatrix} R_{W_{t,t-1,t-2,t-3,t-4}}, V_{V_{t,t-1,t-2t,t-3,t-4}}, \\ V_{W_{t,t-1t,t-2,t-3,t-4}}, R_{O2_{t,t-1,t-2t,t-3,t-4}} \\ E_{CO2_{t,t-1,t-2t,t-3,t-4}}, R_{CO2_{t,t-1,t-2t,t-3,t-4}}, \\ E_{O2_{t,t-1,t-2t,t-3,t-4}}, P_{t-5} \end{bmatrix} \\ y_t = [P_t] \end{cases} \tag{7}$$

Because each sample contains only 1 h data, the kernel size of TCM is set to 1×3. Using the trial-and-error method, the number of layers and channels in the GCN-based model was determined to be three and (32, 64, 128), respectively. Adam-optimizer is selected for gradient descent, and its learning rate was set to 0.001. The regularization coefficient β in Eq. (5) was determined to be 0.2. Additionally, 5% and 10% Gaussian noise was added to the data to investigate the robustness of the model in the case of noisy data.

Table 5
The test RMSE values of baseline models and GCN-based model.

Model	Free noise in test	5% Noise in test	10% Noise in test
SVR [8]	1.298	1.513	1.825
PLS [10]	1.515	1.794	1.985
LSTM [29]	1.279	1.505	1.702
GCN [34]	0.994	1.237	1.593
Proposed GCN	0.552	0.779	1.108

Similar to the previous experiment, SVR [8], PLS [10], LSTM [29], and GCN [34] were chosen for comparison. Moreover, grid search was used to select model hyperparameters for PLS and SVM. The number of layers of LSTM and GCN is set to three, embedding dimension of each layer is (64, 128, 256), and the rest of the settings are consistent with the GCN-based model. The RMSE values of the baseline and GCN-based models are listed in Table 5, with the prediction results shown in Fig. 13. As shown, compared with the SVR and PLS, the RMSE of the GCN-based model in the test of free noise is lower by 57.2% and 63.9%, respectively. Additionally, the prediction performance of GCN in this scenario is better than that of LSTM, which indicates that the importance of variable correlation in soft sensors increases as the system complexity increases and the dataset size decreases. Although a reduced dataset size negatively affects the performance of DL models, the GCN-based model exhibited decent performance with limited data by capturing spatial–temporal localized correlation.

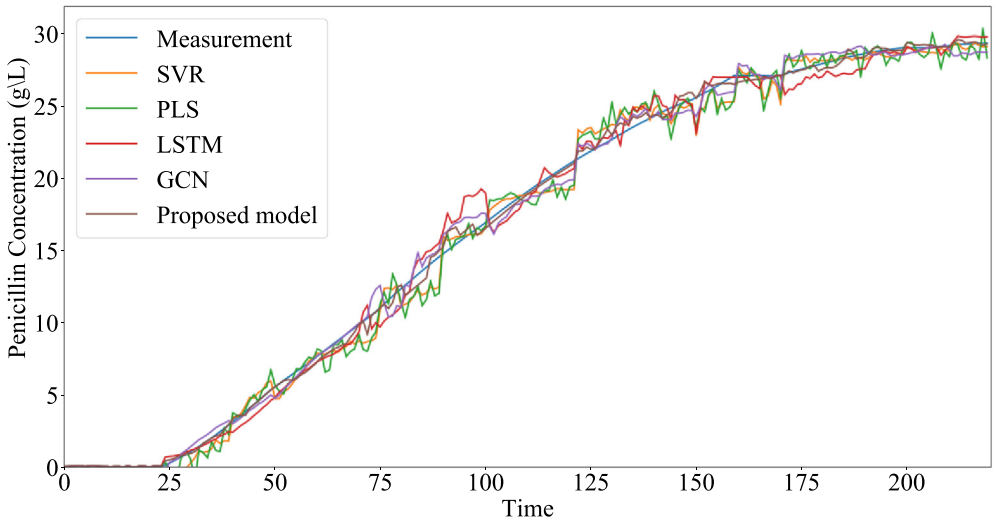
The localized matrix learned by the model is visualized in Fig. 14. Edges with high strength co-existing in the relationship matrix and time correlation matrix should be given greater attention (the edge framed by the red box in Fig. 14). Some examples are: $R_W \rightarrow R_{O2}$, $R_W \rightarrow R_{CO2}$, $R_W \rightarrow V_V$, $R_W \rightarrow V_W$, $V_V \leftrightarrow V_W$, $R_{O2} \rightarrow E_{O2}$, and $R_{CO2} \rightarrow E_{CO2}$. This can be explained in combination with the process mechanism [43]:

(a) R_W changes the solution temperature according to oxygen uptake rate R_{O2} and carbon evolution rate R_{CO2} to control bacterial activity. Meanwhile, R_W affects V_V and V_W by simultaneously changing solution volume and weight.

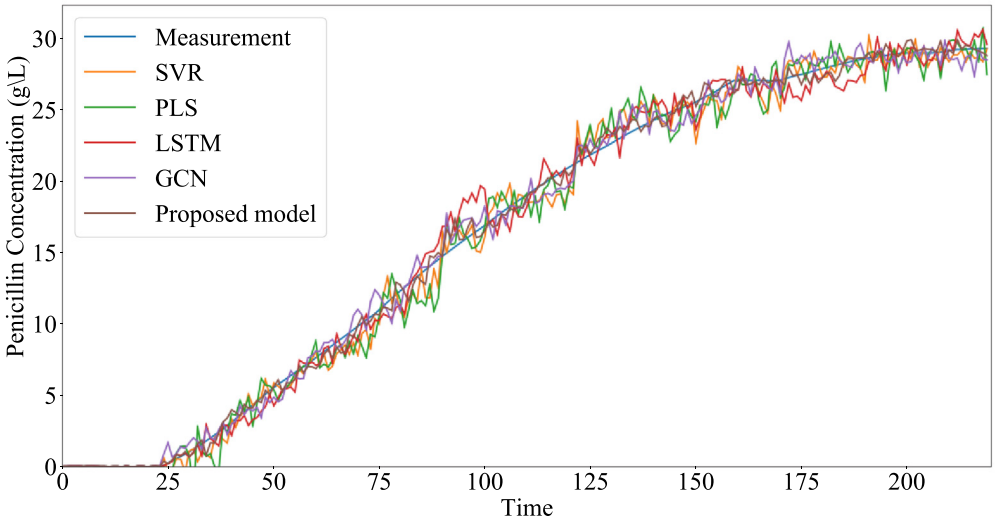
(b) Changes in the oxygen uptake rate R_{O2} and carbon evolution rate R_{CO2} are mainly a result of bacterial respiration and can lead to a significant effect on the concentrations of oxygen E_{O2} and carbon dioxide E_{CO2} in the off-gas.

5. Conclusions

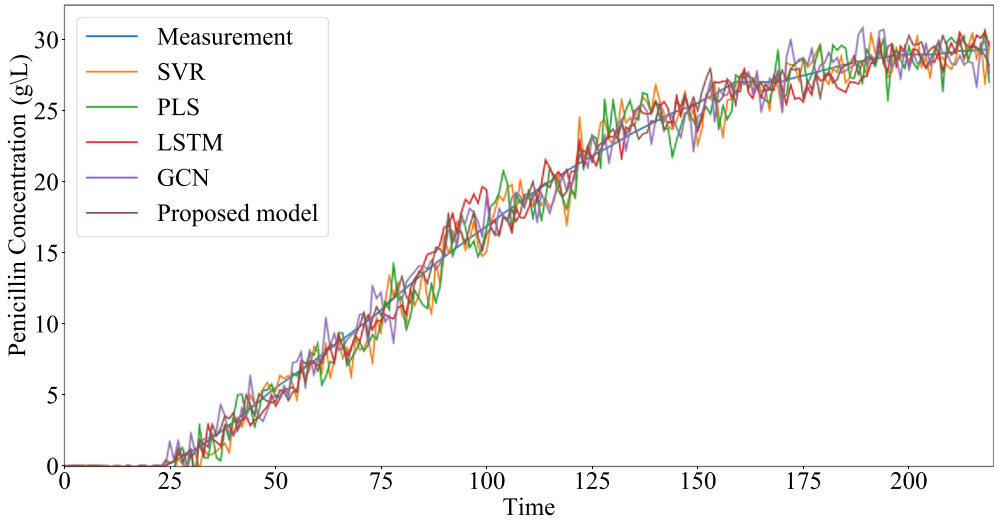
Most data-driven soft sensor models have low explainability. In this work, a GCN-based soft sensor framework is proposed, which exhibits its model superiority by visualizing variable relationships. This model characterizes the cross-correlation among variables by capturing localized spatial–temporal correlations. Variables are used as nodes in the construction of the localized



(a)



(b)



(c)

Fig. 13. Prediction in different scenarios: (a) free noise, (b) 5% noise, and (c) 10% noise.

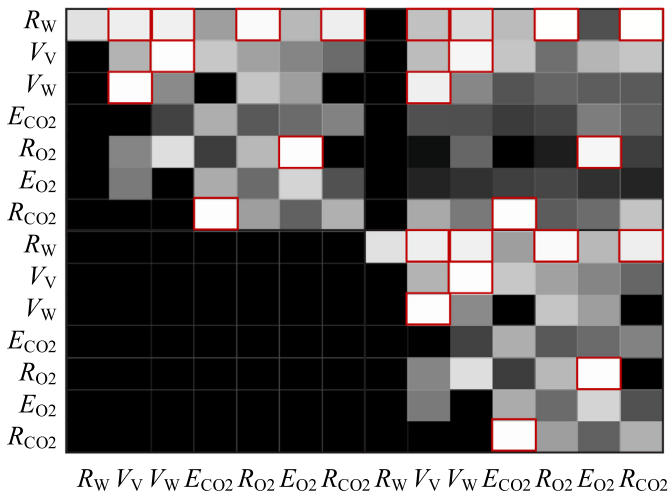


Fig. 14. The localized matrix.

spatial–temporal graph and the network is trained in an end-to-end manner. By stacking multiple spatial–temporal convolution layers, the model learns the underlying relationships and temporal correlations among variables in the form of relationships and localized matrices. Thus, it overcomes the limitations of conventional soft sensors. The case study on the penicillin fermentation process demonstrates the efficacy and superiority of the proposed model. The visualization of localized spatial–temporal correlations among variables demonstrates that the information extracted by the model is consistent with process mechanisms, which demonstrates the superior explainability of the proposed GCN-based mode. However, some limitations still exist in the proposed framework, such as variable relationships should be captured in a dynamic form to fit time-varying processes for further study because these relationships may change as the process progresses.

Nomenclature

ANN	artificial neural network
BN	batch normalization
BNM	bayesian network model
CGC	conditional Granger-cause
DL	deep learning
FCL	fully connected layer
GCN	graph convolutional network
GC test	Granger-cause test
GNN	graph neural network
LSCM	localized spatial–temporal correlation module
LSTM	long short-term memory
MI	mutual information
MIC	maximum information coefficient
PLS	partial least squares
ReLU	rectified linear unit
RMSE	root mean square error
S2S	sequence-to-sequence
STCL	spatial–temporal convolutional layer
SVR	support vector regression
TCM	temporal convolution module
TCN	temporal convolutional network

Symbol

a	the size of convolutional kernel
$\mathbf{A}$	the adjacency matrix of the graph G
$\mathbf{A}$'	the localized matrix of variables
$\mathbf{A}_t$	the time correlation matrix of variables
C	the number of channels
$\mathbf{D}$	the degree matrix of the adjacency matrix $\mathbf{A}$
$\mathbf{E}$	the edge of edges
E_B	the biomass concentration
E_{CO2}	the carbon dioxide concentration
E_{DO}	the dissolved oxygen concentration
E_{O2}	the oxygen concentration in the off-gas
G	the graph
H(.)	the element-level entropy
$\mathbf{j}$, $\mathbf{u}$	two discrete variables
$\mathbf{O}$	the zero matrix
P	the penicillin concentration
$\mathbf{Q}$	the trainable weight of GCN
R_{CO2}	the carbon evolution rate
R_{O2}	the oxygen uptake rate
R_W	the heating/cooling water flow rate
T	the time step of the observation
T_G	the generated temperature
T_m	the number of time step
V	the number of nodes
$\mathbf{V}$	the set of vertices
V_V	the vessel volume
V_W	the vessel weight
V_{SF}	the substrate feed flowrate
$\mathbf{X}_G$	the graph node signal matrix
$\mathbf{X'}_G$	the localized graph signal matrix
β	the coefficient of regularization

CRediT authorship contribution statement

Mingwei Jia: Methodology, Software, Data curation, Visualization, Writing – original draft. **Danya Xu:** Methodology, Software, Validation. **Tao Yang:** Conceptualization, Resources, Writing – review & editing. **Yi Liu:** Conceptualization, Resources, Writing – review & editing, Supervision, Funding acquisition. **Yuan Yao:** Conceptualization, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

The work was supported by the National Natural Science Foundation of China (Grant Nos. 62022073 and 61873241) and National Science and Technology Council, ROC (Grant No. NSTC 111-2221-E-007-005).

References

[1] L. Fortuna, S. Graziani, A. Rizzo, M.G. Xibilia, Soft Sensors for Monitoring and Control of Industrial Processes, Springer, London, UK, 2007.
[2] F.A.A. Souza, R. Araújo, J. Mendes, Review of soft sensor methods for regression applications, Chemom. Intell. Lab. Syst. 152 (15) (2016) 69–79.
[3] B. Lin, B. Recke, J.K.H. Knudsen, S.B. Jørgensen, A systematic approach for soft sensor development, Comput. Chem. Eng. 31 (5/6) (2007) 419–425.
[4] M. Kano, M. Ogawa, The state of the art in chemical process control in Japan: Good practice and questionnaire survey, J. Process Control 20 (9) (2010) 969–982.
[5] S.J. Qin, L.H. Chiang, Advances and opportunities in machine learning for process data analytics, Comput. Chem. Eng. 126 (2019) 465–473.

[6] L.F. Fuentes-Cortes, A. Flores-Tlacuahuac, K.D.P. Nigam, Machine learning algorithms used in PSE environments: A didactic approach and critical perspective, Ind. Eng. Chem. Res. 61 (2022) 8932–8962.
[7] S.B. Chitralekha, S.L. Shah, Application of support vector regression for developing soft sensors for nonlinear processes, Can. J. Chem. Eng. 88 (2010) 696–709.
[8] Z. Li, H.P. Jin, S.L. Dong, B. Qian, B. Yang, X.G. Chen, Semi-supervised ensemble support vector regression based soft sensor for key quality variable estimation of nonlinear industrial processes with limited labeled data, Chem. Eng. Res. Des. 179 (2022) 510–526.
[9] W.W. Yan, D. Tang, Y.J. Lin, A data-driven soft sensor modeling method based on deep learning and its application, IEEE Trans. Ind. Electron. 64 (5) (2017) 4237–4245.
[10] J.H. Zheng, Z.H. Song, Mixture modeling for industrial soft sensor application based on semi-supervised probabilistic PLS, J. Process Control 84 (4) (2019) 46–55.
[11] A. Khosbayar, J. Valluru, B. Huang, Multi-rate Gaussian Bayesian network soft sensor development with noisy input and missing data, J. Process Control 105 (2021) 48–61.
[12] L. Patanè, M.G. Xibilia, Echo-state networks for soft sensor design in an SRU process, Inform. Sci. 566 (2021) 195–214.
[13] K. Desai, Y. Badhe, S.S. Tambe, B.D. Kulkarni, Soft-sensor development for fed-batch bioreactors using support vector regression, Biochem. Eng. J. 27 (3) (2006) 225–239.
[14] P. Kadlec, B. Gabrys, S. Strandt, Data-driven soft sensors in the process industry, Comput. Chem. Eng. 33 (4) (2009) 795–814.
[15] Y.Q. Liu, M. Xie, Rebooting data-driven soft-sensors in process industries: A review of kernel methods, J. Process Control 89 (2020) 58–73.
[16] P. Zhou, W.Q. Chen, C.M. Yi, Z.H. Jiang, T. Yang, T.Y. Chai, Fast just-in-time-learning recursive multi-output LSSVR for quality prediction and control of multivariable dynamic systems, Eng. Appl. Artif. Intell. 100 (2021) 104168.
[17] A. Mohammadi, R. Zarghami, D. Lefebvre, S. Golshan, N. Mostoufi, Soft sensor design and fault detection using Bayesian network and probabilistic principal component analysis, J. Adv. Manu. Process. 1 (4) (2019) 10027.
[18] T.B. Lopez-Garcia, A. Coronado-Mendoza, J.A. Domínguez-Navarro, Artificial neural networks in microgrids: A review, Eng. Appl. Artif. Intell. 95 (2020) 103894.
[19] K.X. Liu, M.K. Zheng, Y. Liu, J.G. Yang, Y. Yao, Deep autoencoder thermography for defect detection of carbon fiber composites, IEEE Trans. Ind. Inform. (2022) http://dx.doi.org/10.1109/TII.2022.3172902, in press.
[20] S. Gao, Y. Dai, Y.J. Li, K.X. Liu, K. Chen, Y. Liu, Multiview Wasserstein generative adversarial network for imbalanced pearl classification, Meas. Sci. Technol. 33 (8) (2022) 085406.
[21] Y. Liu, M.K. Zheng, K.X. Liu, Y. Yao, S. Sfarra, TriMap thermography with convolutional autoencoder for enhanced defect detection of polymer composites, J. Appl. Phys. 131 (14) (2022) 144901.
[22] Q. Liu, Y. Zhang, G. Wu, Z. Fan, Disturbance robust abnormality diagnosis of fused magnesium furnaces using deep neural networks, IEEE Trans. Artif. Intell. (2022) http://dx.doi.org/10.1109/TAI.2022.3168251, in press.
[23] D.X. Chen, X.L. Liu, W.W. Yu, L. Zhu, Q.P. Tang, Neural-network based adaptive self-triggered consensus of nonlinear multi-agent systems with sensor saturation, IEEE Trans. Netw. Sci. Eng. 8 (2) (2021) 1531–1541.
[24] Q.Q. Sun, Z.Q. Ge, Probabilistic sequential network for deep learning of complex process data and soft sensor application, IEEE Trans. Ind. Inform. 15 (5) (2019) 2700–2709.
[25] C. Shang, F. Yang, D. Huang, W.X. Lyu, Data-driven soft sensor development based on deep learning technique, J. Process Control 24 (3) (2014) 223–233.
[26] Y. Liu, C. Yang, Z.L. Gao, Y. Yao, Ensemble deep kernel learning with application to quality prediction in industrial polymerization processes, Chemom. Intell. Lab. Syst. 174 (15) (2018) 15–21.
[27] T. Yang, J.L. Ding, K.G. Vamvoudakis, S.J. Qin, Guest editorial: Industrial artificial intelligence for smart manufacturing, IEEE Trans. Ind. Inform. 17 (12) (2021) 8319–8323.
[28] H.G. Han, Q.L. Chen, J.F. Qiao, An efficient self-organizing RBF neural network for water quality prediction, Neural Netw. 24 (7) (2019) 717–725.
[29] W. Xie, J. Wang, C. Xing, S. Guo, M. Guo, L. Zhu, Variational autoencoder bidirectional long and short-term memory neural network soft-sensor model based on batch training strategy, IEEE Trans. Ind. Inform. 17 (8) (2021) 5325–5334.
[30] X.F. Yuan, B. Huang, Y.L. Wang, C.H. Yang, W.H. Gui, Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE, IEEE Trans. Ind. Inform. 14 (7) (2018) 3235–3243.
[31] S.C. Chang, C.H. Zhao, K. Li, Consistent-contrastive network with temporality-awareness for robust-to-anomaly industrial soft sensor, IEEE Trans. Instrum. Meas. 71 (2021) 2502512.
[32] F. Xia, K. Se, S. Yu, A. Aziz, L.T. Wan, S.R. Pan, H. Liu, Graph learning: A survey, IEEE Trans. Artif. Intell. 2 (2) (2021) 109–127.
[33] L. Shi, Y. Zhang, J. Cheng, H. Lu, Two-stream adaptive spectral graph convolutional networks for skeleton-based action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, 2019, pp. 12026–12035.
[34] F.T.N. Kip, M. Welling, Semi-supervised classification with graph convolutional networks, in: 5th International Conference on Learning Representations (ICLR), Toulon, 2016.
[35] Z.W. Chen, Q. Deng, H. Ren, Z.R. Zhao, T. Peng, C.H. Yang, W.H. Gui, A new energy consumption prediction method for chillers based on GraphSAGE by combining empirical knowledge and operating data, Appl. Energy 310 (15) (2022) 118410.
[36] J.M. Xu, H.B. Ke, Z.W. Chen, X.Y. Fan, T. Peng, C.H. Yang, Over-smoothing relief graph convolutional network-based fault diagnosis method with application to the rectifier of high-speed trains, IEEE Trans. Ind. Inform. 19 (2022) 771–779.
[37] L.J. Feng, C.H. Zhao, Y.L. Li, M. Zhou, C. Fu, Multichannel diffusion graph convolutional network for the prediction of end-point composition in the converter steelmaking process, IEEE Trans. Instrum. Meas. 70 (2020) 1–13.
[38] Y.H. Wang, P.F. Yan, M.G. Gai, Dynamic soft sensor for anaerobic digestion of kitchen waste based on SGSTGAT, IEEE Sens. J. 21 (17) (2021) 19198–19208.
[39] C. Song, Y. Lin, S. Guo, H. Wan, Spatial–temporal synchronous graph convolutional networks: a new framework for spatial–temporal network data forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, (1) New York, 2020, pp. 914–921.
[40] S. Bai, J.Z. Kolter, V. Koltun, Trellis networks for sequence modeling, in: 8th International Conference on Learning Representations (ICLR), New Orleans, 2019.
[41] D.N. Reshef, Y.A. Reshef, H.K. Finucane, S.R. Grossman, G. McVean, P.J. Turnbaugh, E.S. Lander, M. Mitzenmacher, P.C. Sabeti, Detecting novel associations in large data sets, Science 6062 (2011) 1518–1524.
[42] G. Birol, C. Undey, A. Cinar, A modular simulation package for fed-batch fermentation: penicillin production, Comput. Chem. Eng. 11 (2002) 1553–1565.
[43] S. Goldrick, A. Stefan, D. Lovett, G. Montague, B. Lennox, The development of an industrial-scale fed-batch fermentation simulation, J. Biotechnol. 193 (1) (2015) 70–82.
[44] C. Zheng, B. Zheng, W. Cheng, D.J. Song, J.C. Ni, W.C. Yu, H.F. Chen, W. Wang, Robust graph representation learning via neural sparsification, in: 37th International Conference on Machine Learning, ICML, 2020, pp. 11458–11468.
[45] H. NT, T. Maehara, T. Murata, Revisiting graph neural networks: graph filtering perspective, in: 25th International Conference on Pattern Recognition, ICPR, 2021, pp. 8376–8383.