TU Delft

Delft University of Technology

Document Version
Final published version

Licence
CC BY

Citation (APA)
Theisen, M. F., Meesters, G. M. H., & Schweidtmann, A. M. (2026). Graph neural networks for soft sensors: Learning from process topology and operational data. *Computers and Chemical Engineering*, *206*, Article 109532. https://doi.org/10.1016/j.compchemeng.2025.109532

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Computers and Chemical Engineering 206 (2026) 109532

Contents lists available at ScienceDirect

Computers and Chemical Engineering

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

Graph neural networks for soft sensors: Learning from process topology and operational data

Maximilian F. Theisen, Gabrie M.H. Meesters, Artur M. Schweidtmann *

Department of Chemical Engineering, Delft University of Technology, Delft, 2629 HZ, Netherlands

ARTICLE INFO

Keywords:
Process topology
Digital twins
Process operations
Dynamic modeling
Deep learning
Graph neural networks

ABSTRACT

Soft sensors estimate process variables that are difficult or impossible to measure directly by using mathematical models and available sensor data, e.g., product concentrations. Machine learning-based approaches have become popular for soft sensing tasks. These approaches offer automatic modeling using historical process data but lack basic process information, such as the process topology. This can lead to (1) modeling of correlations instead of causation between process measurements, (2) model deterioration in deployment due to unseen process scenarios, and (3) large data requirements. To overcome these shortcomings, we propose a novel ML modeling approach incorporating the process topology into soft sensor models for improved spatio-temporal modeling. For this, we propose process topology-aware graph neural networks. We combine process topology and sensor data by representing process data in a directed graph and leverage these process graphs to train graph neural networks. Our method demonstrates enhanced model robustness, reduced data requirements, and more intuitive data representations compared to standard black-box machine learning modeling approaches. Overall, this work introduces a new paradigm for soft sensing by directly embedding process information into the data, paving the way for more efficient and reliable digital twin applications.

1. Introduction

Soft sensors play an important role in modern (bio-)chemical process operations because they allow the real-time estimation of process parameters that are hard to measure inline, e.g., product yields (Yi et al., 2020) or substrate concentrations (Salgado et al., 2004). Soft sensors leverage available information, such as hard sensors in the plant, to estimate their targets (Luttmann et al., 2012). To provide value in process operations, soft sensors require accurate dynamic models of the underlying plant. Towards this end, machine learning (ML) based models have received much attention in literature (Jiang et al., 2021b) because ML models learn statistical correlations from historical process data in an end-to-end fashion.

One common approach to leverage ML models for soft sensors is to use flat, tabular data-based regression models (Souza et al., 2016; Jiang et al., 2021c; Kadlec et al., 2009; Curreri et al., 2021; Lin et al., 2007; Fortuna et al., 2007). In tabular data models, all input variables are processed as individual, independent features without any predefined relational structure. During training, the ML model learns statistical correlations among the input variables to predict the soft sensor target (Sun and Ge, 2021). Many different ML models have been explored towards this end. Tree-based models, such as random forest regression and extreme gradient boosting (XGBoost) have been utilized in several works (Ching et al., 2022; Cheng et al., 2023). Besides, support vector regression-based approaches have also been explored (Kaneko and Funatsu, 2014; Yan et al., 2004). Finally, (deep) neural network-based regression learners have been extensively developed (Shang et al., 2014) as well due to their versatile nature.

The pure, data-driven nature of ML models without any underlying process awareness can lead to challenges in developing and deploying soft sensors in industrial settings (Cao et al., 2022). Among them, the following three challenges are directly related to the inherent statistical learning nature of ML-based soft sensors. (1) Spurious relationships refer to correlations detected by the model between sensor readings and outcomes that do not reflect any true causal link, but rather are coincidental or influenced by hidden variables. These spurious relationships can lead to erroneous predictions and unreliable performance in real-world applications (Wang et al., 2013; Klaeger et al., 2021). (2) Covariate shifts occur when the statistical properties of the input variables change over time, leading to a mismatch between the data the model was trained on and the data it encounters during operation (Zhao et al., 2024; Lu et al., 2020). This can result in degraded model performance and inaccurate predictions as the model fails to generalize to the new data distribution. As denoted by Souza et al. soft sensor models are

* Corresponding author.
E-mail address: a.schweidtmann@tudelf.nl (A.M. Schweidtmann).

https://doi.org/10.1016/j.compchemeng.2025.109532
Received 3 July 2025; Received in revised form 2 December 2025; Accepted 13 December 2025
Available online 24 December 2025
0098-1354/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

therefore required to be maintained regularly in industry (Souza et al., 2016). Covariate shifts can have many causes in process operations, e.g., catalyst deactivation, drifts in sensor measurements over time or simply setpoint scenarios not encountered in the training data. (3) Data quantity is yet another consideration when building ML applications, such as soft sensors (Grimstad et al., 2023; Zhuang et al., 2022). Modern ML algorithms require large amounts of high-quality data to perform accurately. Procuring this data however can be time intensive and cost intensive as denoted by Bortz et al. (2023). The pure, statistical learning of ML models thus has drawbacks when applied to soft sensor development.

Hybrid models can overcome some of the drawbacks of data-driven ML models but require additional mechanistic insights (Schweidtmann et al., 2024; Daoutidis et al., 2024; Venkatasubramanian, 2018; Bradley et al., 2022; von Stosch et al., 2014). Hybrid modeling frameworks, also referred to as grey-box models, aim to combine pure, data-driven ML models with mechanistic insights by directly incorporating physical models thereby overcoming some of these drawbacks. This helps in enforcing physical predictions, increase transparency, and ultimately trust. Hybrid models require mechanistic models describing the system, which increases the overall modeling effort and may not be always feasible. Furthermore, the ML models used in hybrid modeling are still black box models, unaware of the underlying process.

Towards inherently more process-aware, context-oriented ML models, recent literature has explored how chemical processes can be represented in appropriate information representations to be communicated to ML models. Two types of representations have been commonly leveraged, graph-based and string-based process representations (Gao and Schweidtmann, 2024). In both representations, the process topology is encoded to make the ML model learn about the context of the process. In graph-based representations, the process topology is represented as a graph with unit operations nodes and streams as edges. Alternative, more holistic graph-based information representations have also been proposed, such as (py-)DEXPI (Theissen et al., 2021; Goldstein et al., 2025) or OntoCape (Marquardt et al., 2010) as well as different semantic representations such as hypergraphs (Mann et al., 2024) or knowledge graphs. Graph neural networks (GNNs) are used to learn from process graphs (Stops et al., 2022; Gao and Schweidtmann, 2024; Oeing et al., 2023; Balhorn et al., 2024). In sequence-based representations, the process topology is expressed in a string-based format by creating a sequence of the process graph (d'Anterroches, 2006; Vogel et al., 2023; Mann et al., 2024; Li, 2024). This allows to leverage transformer-based models (Vaswani et al., 2017) or Recurrent Neural Networks (Hochreiter and Schmidhuber, 1997) for training. Several sequence-based representations have been proposed, such as SFILES (d'Anterroches, 2006) and its extensions SFLIES2.0 (Vogel et al., 2023) and eSFILES (Mann et al., 2024), as well as phenomena based string representations as proposed by Li (2024).

GNNs have also gained attention in the context of ML-based soft sensor models. Lin et al. (2024) applied a graph attention network together with a long short term memory network to model the spatio-temporal dependencies within a propylene plant. In their graph modeling approach, the graph is constructed by considering each sensor as a node that is connected to all other nodes. Huang et al. (2021) developed a graph sensing neural network-based approach for wafer production. In their approach, they used GNNs to consider both textual and numerical information. Their graph representation is learned by the network. Recently, Allen and Cordiner (2024) developed a forecasting model for a wastewater treatment plant. They developed a disentangled graph convolution approach to extract spatial dependencies between sensor measurements and modeled the sensor measurements as a fully connected graph. Niresi et al. (2024) build a sensor network-based on real sensors as well as physics-based nodes to generate enhanced graphs for soft sensing tasks. Related to soft sensing, Wang et al. built causal GNNs to detect faults in process operation (Wang et al., 2024). This also builds on previous work by Hu et al. (2018), who proposed a framework to leverage process connectivity information in alarm management systems. Previous works demonstrate the applicability of GNNs to soft sensors well, it remains open how the process topology along with sensor data can be leveraged.

Recognizing the advances in the application of GNNs for soft sensing tasks in industrial processes, we aim to extent the GNN-based modeling approach by holistically embedding information about the process topology into the GNN modeling framework, such as sensor types, sensor locations, unit operation types and flow directionalities. In this way, we aim to make the GNN model more aware of the entire plant topology. We take a three step approach: First, we develop a soft sensor modeling approach using GNNs to incorporate the process topology. We are the first to explicitly incorporate the process topology into the model using a data representation that represent the entire process topology. Second, we introduce three case studies that are challenging to current state-of-the-art (tabular) ML approaches to soft sensor modeling. Third, we benchmark the topology-aware GNN approach against common ML models including artificial neural networks, random forest regression, and XGBoost.

The remainder of the manuscript as follows. In Section 2, we explain the ML modeling tools used in this work. In Section 3, we introduce our developed topology-aware GNN soft sensor. In Section 4, we discuss the case studies that we use to demonstrate properties of the topology-awareness. We present and discuss our results in Section 5 and benchmark our approach against current ML models. To further investigate those the modeling mechanisms of the GNN model, we carry out several ablations in Section 6 We conclude our work in Section 7.

2. Background

In this section, we introduce the two modeling tools for spatio-temporal modeling, GNNs and transformer encoders.

2.1. Graph neural networks

GNNs are a class of neural networks designed to handle graph-structured data. Unlike traditional neural networks, which operate on fixed-size inputs, GNNs can process data where the relationships between elements are best represented as a graph. Each node in the graph represents an entity, and the edges represent relationships or interactions between these entities. GNNs are used in various applications, including social network analysis, recommendation systems, and biological network modeling (Kipf and Welling, 2017). GNNs have also been a popular modeling technique in computational chemistry and chemical engineering in different applications, e.g., for predicting the products of organic reactions (Coley et al., 2019), novel crystal material discovery (Jiang et al., 2021a) or estimating fuel ignition qualities-based on molecular graphs (Schweidtmann et al., 2020).

At the core, the GNNs iteratively update the representation of each node by aggregating information from its neighbors, see Fig. 1. This process, known as *message passing*, allows nodes to integrate information from their local graph structure. The updated representation of a node after k iterations can capture information from nodes that are up to k hops away in the graph. Mathematically, this is expressed as:

$$\mathbf{h}_v^{(k)} = \text{UPDATE}\left(\mathbf{h}_v^{(k-1)}, \text{AGGREGATE}\left(\{\mathbf{h}_u^{(k-1)} \mid u \in \mathcal{N}(v)\}\right)\right)$$

where $\mathbf{h}_v^{(k)}$ is the representation of node v at iteration k, $\mathcal{N}(v)$ denotes the neighbors of v, and UPDATE and AGGREGATE are functions that combine the information of the node and its neighbors (Gilmer et al., 2017). Common aggregation functions include sum, mean, and max, while the update function is typically a neural network (Battaglia et al., 2018). A common choice is a gated recurrent unit (GRU), that considers the current hidden state $\mathbf{h}_v^{(k-1)}$ and the aggregated message $\mathbf{m}_v^{(k)} = \text{AGGREGATE}\left(\{\mathbf{h}_u^{(k-1)} \mid u \in \mathcal{N}(v)\}\right)$:

$$\mathbf{h}_v^{(k)} = \text{GRU}\left(\mathbf{h}_v^{(k-1)}, \mathbf{m}_v^{(k)}\right)$$

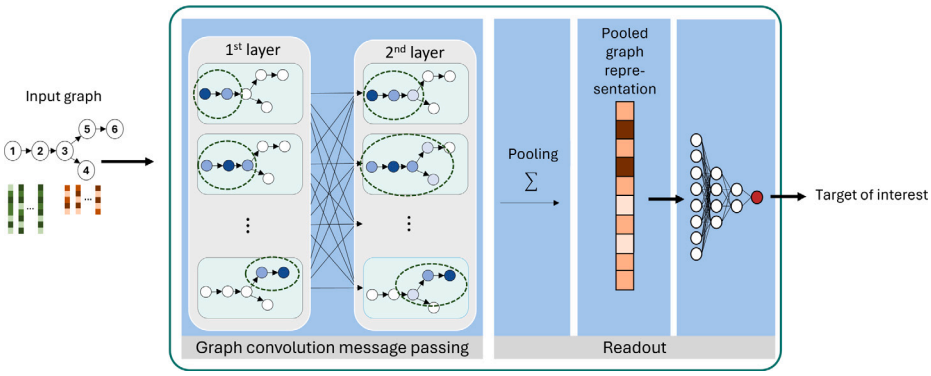


Fig. 1. Illustration of a GNN workflow, adapted from Schweidtmann et al. (2020). The left section shows input node features. The middle section demonstrates the graph convolution message passing through multiple layers, where each layer aggregates information from neighboring nodes. The right section shows the pooling operation that combines node representations into a single graph-level representation, which is then used for readout to predict a target of interest. This could be for instance for a classification or for a regression task.

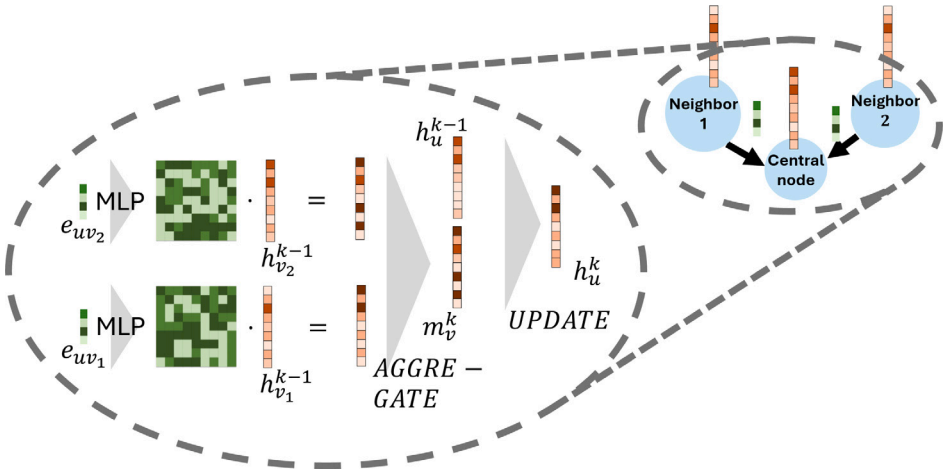


Fig. 2. The message passing internals visualized for a node u with two neighbors as described in Gilmer et al. (2017).

The edge information of the connected nodes $\mathbf{e_{uv}} \in \mathcal{N}$ can also be incorporated into GNNs to enhance the model's capabilities. This is achieved by including edge features in the message passing process. The edge features $\mathbf{e}_{uv}$ associated with the edge between nodes u and v are utilized before the aggregation step. The entire message passing for node $\mathbf{v}$ can then be described as:

$$\mathbf{m}_v^{(k)} = \text{AGGREGATE}\left(\{\Theta(\mathbf{h}_u^{(k-1)}, \mathbf{e}_{uv}) \mid u \in \mathcal{N}(v)\}\right)$$

$$\mathbf{h}_v^{(k)} = \text{UPDATE}\left(\mathbf{h}_v^{(k-1)}, \mathbf{m}_v^{(k)}\right)$$

Hereby Θ is a multi-layer perceptron (MLP) that processes the edge information $\mathbf{e}_{uv}$ with the corresponding node embedding $\mathbf{h}_u^{(k-1)}$. In this formulation, the aggregation function takes into account both the node features and the edge features, which allows the model to learn more nuanced representations that reflect both node attributes and the relationships between nodes. The full message passing for a single node with two neighbors is visualized in Fig. 2. For directed graphs, the edge direction can be modeled as well. In the aggregation step, only nodes are that are directed towards v are considered to be neighbors $\mathcal{N}(v)$. This directedness allows to control the flow of information in the GNN. The expressive power of GNNs stems from their ability to capture dependencies in graph-structured data. By stacking multiple layers of message passing, GNNs can learn hierarchical representations of graphs. This is particularly useful in tasks where the graph structure is critical for prediction (Xu et al., 2018).

2.2. Transformer encoders

Transformers have been a key enabling technology in recent natural language processing advances such as language translation (Stahlberg, 2020), text summarizing (Liu and Lapata, 2019), and text classification (Shaheen et al., 2020). Moreover, transformers have recently been introduced to computer vision related tasks (Dosovitskiy et al., 2021). Transformers are a type of neural network architecture that operates on sequential data. They exhibit several favorable properties, such as their inherent ability to model long range dependencies, their versatility and their efficient, parallel training procedure. They have become the popular choice for many tasks involving sequential data, often substituting recurrent neural networks such as GRUs (Dey and Salem, 2017) or long short-term memory (LSTM) networks (Yu et al., 2019).

The transformer encoder generates an encoded sequence from an input in a single pass. It breaks the input sequence into tokens, the smallest units. It then processes the input with repeated layers, each comprising two primary components (Fig. 3). The first component is multi-head attention, which allows the model to simultaneously focus on different parts of the sequence, capturing various dependencies and relationships between tokens. The second component is an MLP that applies a series of transformations to further process the data. By

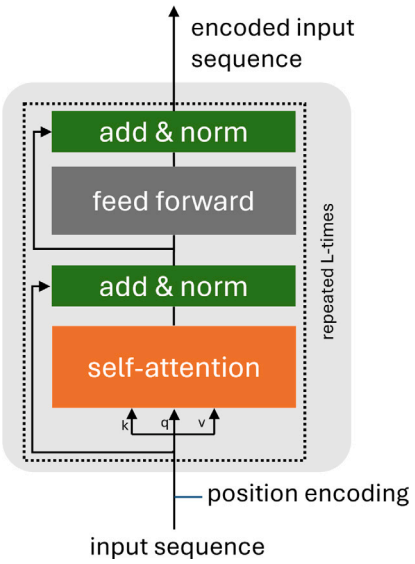


Fig. 3. Illustration of a transformer encoder architecture.

iteratively applying these layers, the encoder transforms the original sequential input into a detailed and meaningful encoded representation. To add positional context into the encoded representation, a positional embedding is added to the input sequence before being fed to the encoder.

Within the encoder, a key component is the attention mechanism (Vaswani et al., 2017). Attention allows the model to weigh the importance of a token towards all other tokens in a sequence. For that, a query $\mathbf{q} \in \mathbf{Q}$, key $\mathbf{k} \in \mathbf{K}$, and value $\mathbf{v} \in \mathbf{V}$ vector for each token are created. The attention is then defined as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{1}$$

where the dot-product product of the query vectors $\mathbf{Q}$ and the key vectors $\mathbf{K}$ is a measure of relatedness. Then, the value vector is scaled with this dot product and d_k is the dimension of the query and key. There are two major advantages of attention over previous methods. First, compared with hidden state-based methods such as LSTMs, the attention mechanism can capture the relative importance of each token over a long distance (Le and Zuidema, 2016). Second, the training process can be parallelized, leveraging advances in GPU computing and thus making the training feasible and efficient for very large datasets (Liu et al., 2020).

Transformers have been recently leveraged to model time-series data, outside and within chemical engineering. Outside of chemical engineering, time-series transformers have shown promising results for different tasks, such as time series forecasting, anomaly detection and time-series classification (Wen et al., 2022). Within chemical engineering, transformers have been investigated in different applications for modeling and control of dynamic systems. Sitapure et al. developed soft sensors and MPC-based control utilizing time-series transformers for modeling on crystallization systems (Sitapure and Kwon, 2023b,a; Sitapure and Sang-Il Kwon, 2023; Sitapure and Kwon, 2024). Arce Munoz et al. leveraged the transformer architecture for transfer learning across dynamical systems (Arce Munoz et al., 2024) and applied it in a thickener control case study (Arce Munoz and Hedengren, 2025). In statistical process control, transformers have also been utilized for anomaly detection (Zhu et al., 2024) and for early fault prognosis through time series forecasting (Bai and Zhao, 2023a,b). Further, Lastrucci et al. utilized time series transformer for solving reactor design ordinary differential equations (Lastrucci et al., 2024). These works demonstrate the applicability and utility of time-series transformers for various chemical engineering tasks.

3. Topology-aware soft sensors

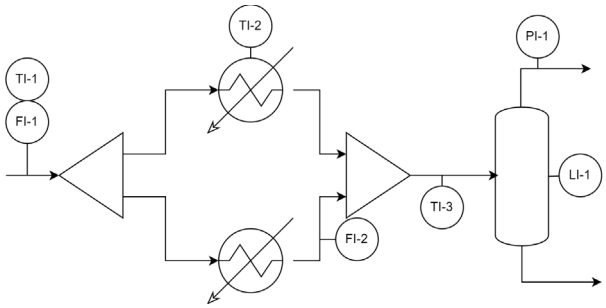
The system dynamics of chemical processes depends on many factors including the process topology, thermodynamic states, flow rates, concentrations, design and operating variables, components, and time dependencies. Representing this information is crucial for effective learning and generalization of the ML model. Similar to previous works (Lin et al., 2024), we divide the modeling task into a spatial and a temporal modeling component.

3.1. Modeling spatial relationships

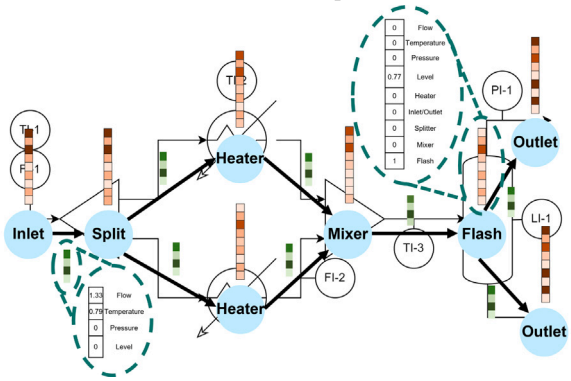
To account for the spatial relationships, we represent the underlying process as a structured graph, following the topology of the underlying process. We model three components of the process topology, (1) the unit operations, (2) how they are connected to each other, and (3) the affiliated sensor data. Unit operations, such as mixing, separation or reaction, are modeled as nodes of the graph. To account for the type of unit operation, we one hot encode the type as an attribute in the node vector. The streams are the corresponding edges in the graph. Streams connect unit operations and controllers, and we represent them as edges in the graph. The direction of flow for streams is represented as the direction of edges in the graph. Recycle streams are also represented with directed edges, introducing cyclical structures into the topology graph. Sensor data is embedded directly into the attribute vectors of the corresponding graph elements, encoding both the measurement type and sensor location within the plant. Specifically, data from unit operations is integrated into node attributes, while data from streams is incorporated into edge attributes. The sensor data is embedded in the attribute vector of the corresponding graph element. We chose this way of representing the flowsheet as a graph for three reasons. First, the representation is compact because the node and edge embeddings are densely packed representations. This keeps the number of nodes per graph low, potentially reducing issues with over-smoothing during message passing (Rusch et al., 2023; Hamilton, 2020a). Second, this information representation allows the application of homogeneous GNNs, which are oftentimes favorable for simplicity, scalability and efficiency (Wang et al., 2023). Third, the representation is similar to successfully utilized flowsheet representations, e.g., for steady-state flowsheet representations (Stops et al., 2022).

To illustrate our information representation approach, we consider the flowsheet as shown in Fig. 4. The process consists of a feed which is split and heated. The two streams are then mixed again and sent to a flash vessel, with a top and a bottom outlet. Further, many sensor measurements are taken both in the unit operations and the streams. In a first step, we model the unit operations as nodes. In their node embeddings, the type of the unit operation is added as a one hot encoding as well. In the second step, we model the streams as directed edges of the graph. In the third step, we add the sensor measurements as further attributes to the corresponding nodes and edges. The resulting node embedding in this example is of size nine, since we have five unit types (inlet, splitter, heater, flash, outlet) and four measurement types (flow, level, temperature, pressure). The resulting edge embedding is of size four, since we only need to include the measurement types.

For the aggregation phase in the GNN, we consider and compare two modeling approaches. In the first mode, we aggregate by extracting the node embedding of the unit operation where the soft sensor target variable is located. This approach follows spatial modeling by only considering the local information aggregated at the target node or edge. From a graph representation learning perspective, this corresponds to a node-level regression task (Hamilton, 2020b). In the second mode, we aggregate by pooling all node embeddings of the graph. We consider



(a) Example process flowsheet with several unit operations and sensor measurements.



(b) Process graph representation laid over the original flowsheet.

Fig. 4. Combined view of example tabular data and process representation as a graph.

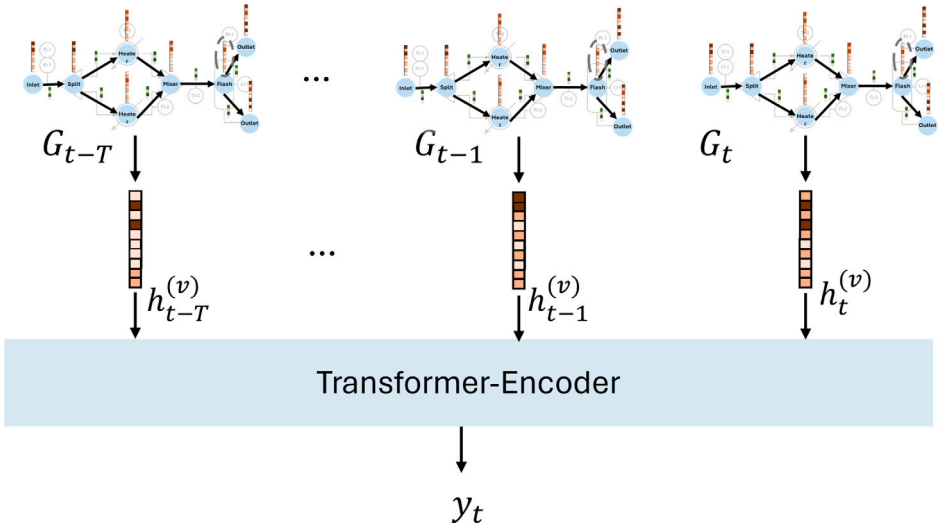


Fig. 5. Time-series modeling illustrated to predict target y_t with a look back window of size T. First, for each time step t_i an embedding is created with a GNN. This embedding is then fed to the transformer encoder for the final prediction of y_t.

the type of pooling as a hyperparameter. We thus aggregate information from unit operation far away from the soft sensor target unit operation as well. From a graph representation learning perspective, this corresponds to a graph-level regression task (Hamilton, 2020b).

3.2. Temporal modeling with transformer

For modeling the temporal component in process operations, we deploy a transformer architecture to process node embeddings across a look back horizon T as a sequence over the GNN embeddings. This approach is illustrated in Fig. 5. Hereby, T is a hyperparameter that determines the size of the look back horizon and is problem dependent. For very quick processes ($T = 1$), we observed that skipping the encoder block improves model performance by eliminating the redundant temporal modeling stage.

The model operates as follows to predict the soft sensor target(s) y_t at time t:

1. **Node embedding generation:** At each time step t_i in the look back horizon, a GNN is used to generate node embeddings $h_i^v \in$

Table 1
Sensor data available for pump network process, used as the input to the soft sensor models.

Sensor type	Location	Sensor type	Location
Pressure	Feed	Temperature	Feed
Pressure	Stream S2	Temperature	Stream S2
Pressure	Stream S4	Temperature	Stream S4
Flow	Pump P-101	Flow	Pump P-102

$\mathbb{R}^{V\times H}$ (with V and H being the number of nodes and their hidden dimension, respectively) for each node v in the graph of that time step G_{t_i}:

$$h_i^v = \mathrm{GNN}(G_{t_i}) \tag{2}$$

2. **Sequence formation:** The embeddings from each time step are arranged in a temporal sequence $S_T = \{h_t^v, h_{t-1}^v, \ldots, h_{t-T}^v\} \in \mathbb{R}^{H\times T}$.
3. **Mode-dependent embedding selection:** The choice of embeddings depends on the prediction mode m as previously outlined:
   - For node-level prediction, the embeddings of the target unit operation's node v_{target} are used:

$$S_t = \{h_t^{v_{\text{target}}}, h_{t-1}^{v_{\text{target}}}, \ldots, h_{t-T}^{v_{\text{target}}}\} \tag{3}$$

$$S_t = \{\mathrm{POOL}(h_t), \mathrm{POOL}(h_{t-1}), \ldots, \mathrm{POOL}(h_{t-T})\} \tag{4}$$

   where POOL is an aggregation function (e.g., mean, max, or sum) applied to all node embeddings h_i at each time step.
4. **Transformer processing:** The time sequence S_t is processed by a transformer encoder model $f_{\text{transformer}}$:

$$S_t^{\text{reduced}} = f_{\text{transformer}}(S_t) \tag{5}$$

5. **Target Prediction:** The transformer's output $S_t^{\text{reduced}} \in \mathbb{R}^H$ is used to predict the final target quantities y_t at time step t specific to the soft sensor application with an MLP regressor.

$$y_t = f_{\text{MLP}}(S_t^{\text{reduced}}) \tag{6}$$

This approach combines the spatial relationship modeling capabilities of GNNs with the sequential modeling strengths of transformers. The flexibility in choosing between node-level and graph-level predictions, as well as the ability to adjust the aggregation method for graph-wide embeddings, allows this model to be adapted to various process characteristics and prediction tasks. Using a look back horizon, the proposed approach allows to model dynamics by using large T. This way, the proposed model can capture dependencies between inputs with time lag and the soft sensor target at the current time step.

4. Case studies

We demonstrate our approach on three illustrative case studies: (1) A pump network, (2) a blending network, and (3) an ammonia synthesis loop. The three case studies are chosen to illustrate distinct challenges of soft sensor modeling. In the first two case studies, the operating regime changes at test time. In the first case in Section 4.1, this causes a covariate shift. The second case study Section 4.2 is designed so that spurious relationships between predictors and target may arise due to common hidden variables. The third case study Section 4.3 is a large scale dynamic simulation of an ammonia synthesis loop.

Table 2
Sensor data of the blend network process, used as the input to the soft sensor models.

Sensor type	Location	Sensor type	Location
Pressure	Stream S1	Temperature	Stream S1
Flow	Stream S1	Pressure	Stream S2
Temperature	Stream S2	Flow	Stream S2
Pressure	Blend 1 outlet	Temperature	Blend 1 outlet
Flow	Blend 1 outlet	Pressure	Stream S4
Temperature	Stream S4	Flow	Stream S4
Pressure	Stream S5	Temperature	Stream S5
Flow	Stream S5	Pressure	Stream S6
Temperature	Stream S6	Flow	Stream S6
Pressure	Stream S7	Temperature	Stream S7
Flow	Stream S7	Pressure	Stream S8
Temperature	Stream S8	Flow	Stream S8
Pressure	Stream S9	Temperature	Stream S9
Flow	Stream S9	Pressure	Blend 2 outlet
Temperature	Blend 2 outlet	Flow	Blend 2 outlet
Concentration	Water in Blend 2	Flow	Pump P-101
Flow	Pump P-102	Flow	Pump P-103

4.1. Covariate shift: Predicting flows in a water-pump network

The first case study investigates the influence of the covariate shift through setpoint changes. We investigate this covariance shift in an illustrative pump network (shown in Fig. 6), where a feed stream is split into two parallel streams in unit S-101. For both streams, a pump, P-101 and P-102 respectively, increase the pressure. Afterward, the parallel streams are mixed again in mixer M-101. The speed of both pumps are controlled with FC-1 and FC-2 to set the total flow. The target of the soft sensor model is the product flow. To model the product flow, the sensor measurements shown in Table 1 are available. Accordingly, there are three pressures, three temperatures and two flows as shown in Table 1 as inputs to the model, each for the past T time steps. The output of the model is the estimated total flowrate at the current time step.

The flow through both parallel streams is shown in Fig. 7. During normal operation (train and validation data set), the setpoint of FC-2 is kept constant, while the setpoint of FC-1 is altered, causing the flow through stream S1 to change and S3 to be constant. For testing, we differentiate two modes, Mode I and Mode II. In Mode I (Fig. 7(a)) the flow rate through P-102 is constant, while the flow rate through P-101 is altered, similar to the preceding operation. In Mode II (Fig. 7(b)) the setpoint of FC-2 is altered as well, leading to a variation of flow through P-102.

Mode II poses a challenge for ML models because, under normal operation, P-102's flow rate is constant, but it changes during testing. ML models rely on variation in training data to learn input–output relationships. Since there is no variation in the training data with respect to how P-102 is operated, there is little to learn for ML models about how P-102 behaves. At test time, P-102 is operated differently, creating a out of training data distribution event.

4.2. Spurious relationships: Predicting product concentration in a blending network

The second case study investigates the influence of spurious relationships for soft sensor models by modeling a blending process, see Fig. 8. There are three tanks T-101, T-102, and T-103 containing water and ethanol. The process aims to mix those to create two blends with different ethanol concentrations. For this, the ethanol stream coming from T-102 is split and mixed with the respective water streams from T-101 and T-103. To adapt the water/ethanol ratio in both blends, each feed pump is controlled with flow controllers FC-1, FC-2, and FC-3. The soft sensor target is to predict the concentration of water in Blend 1. For this, the sensor measurements shown in Table 2 are available. In particular, the measurements also include the water concentration

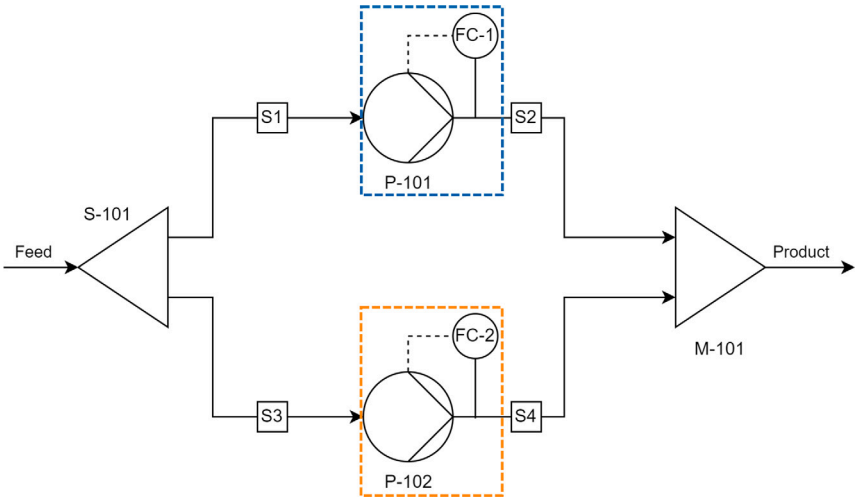
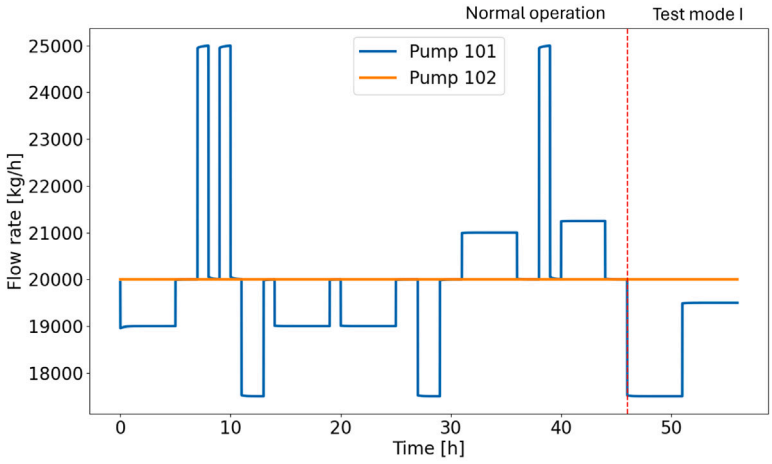
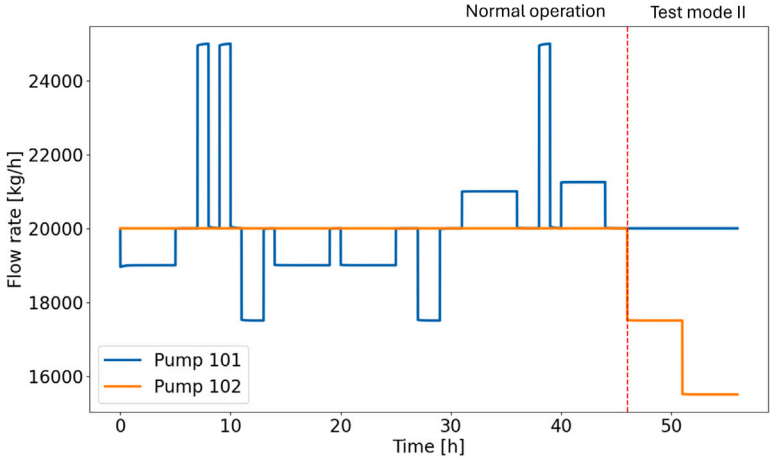


Fig. 6. Flowsheet of the pump network case study.



(a) Mode I: At test time, the operation is similar to the training time



(b) Mode II: At test time, the flow through P-102 deviates from its constant rate during normal operation.

Fig. 7. The flow through the two parallel streams over time, for both test Mode I and test Mode II.

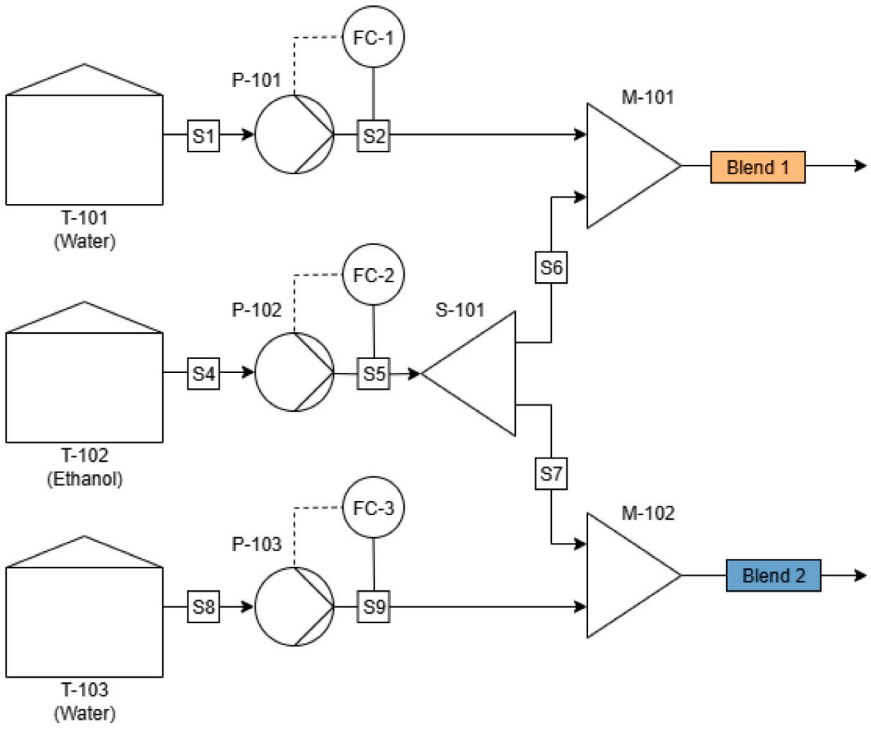


Fig. 8. Flowsheet of the blending network case study.

in Blend 2. The concentration of water in the two outlet streams is shown in Fig. 9. During normal operation (train and validation data set), the setpoints of FC-1 and FC-3 are kept constant, while the setpoint of FC-2 is altered, causing the concentration of water in the blends to change simultaneously. The inputs to the model are thus nine flows, nine temperatures, twelve flows and one concentration, each for the past T time steps. The output of the soft sensor model is the estimated concentration of water in Blend 1 at the current time step.

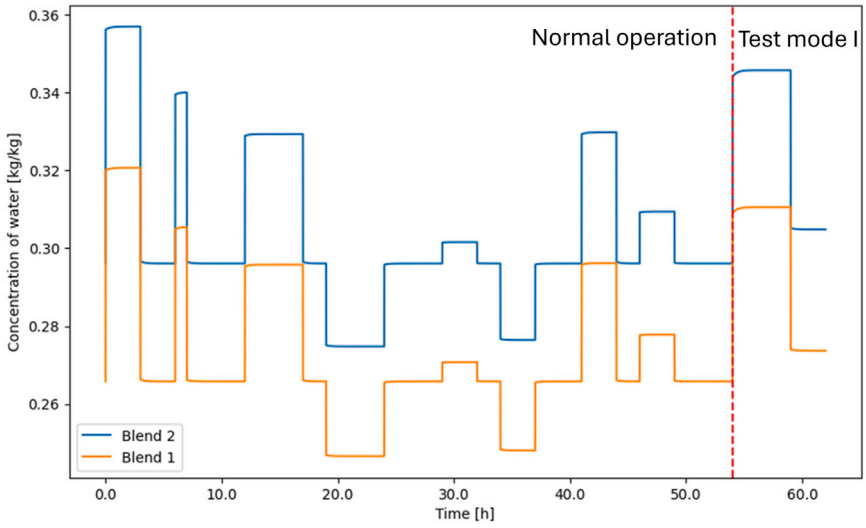
We again differentiate between two modes at test time. Mode I follows the same operation as the training set, keeping the correlation between the two blends intact. In Mode II, during abnormal operation (test data set), the setpoint of FC-1 is altered. This causes the composition of Blend 1 to change, while Blend 2 remains mostly unchanged. Mode II is challenging for ML models because the concentrations of water in Blend 1 and 2 are highly correlated during normal operation but not at test time. ML models cannot differentiate between correlation and causality. Thus, these models identify a correlation between the concentration of water in Blend 1 and Blend 2. However, this correlation is caused by the ethanol stream feeding both blends. When this relationship is broken at test time, the ML models will likely not realize this and predict wrong concentrations instead.

4.3. Predicting the product concentration in an ammonia synthesis loop

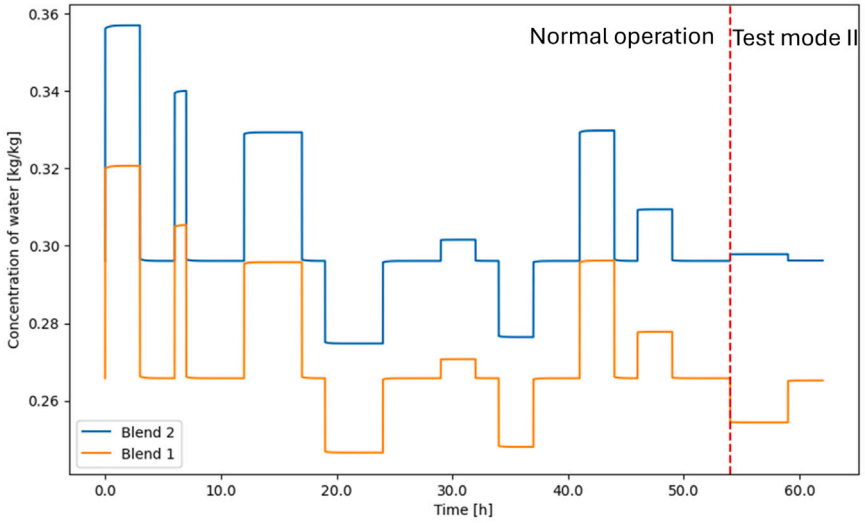
As a case study with a larger process topology, we investigate the soft sensor development for the prediction of ammonia in an ammonia synthesis loop. The process has been previously developed and investigated by Araújo and Skogestad (Araújo and Skogestad, 2008) and is visualized in Fig. 10. In the process, a feed consisting of nitrogen (N2) and hydrogen (H2) as well as impurities of methane and argon is compressed in Compressor K-101. It is then mixed with the reactor outlet and flashed in Vessel V-101 to separate the ammonia from the stream. The liquid phase leaves V-101 as the product stream. The gaseous phase is purged to remove inert ammonia and methane. It is then further compressed in Compressor K-102, heated and fed to the reactor R-101 with two quench stages. After leaving R-101, the stream is cooled and again mixed with the fresh feed. The control structure for the dynamic operation follows the structure outlined in the original work: The process is controlled using 4 PID controllers. The fresh feed is controlled using flow controller FC-1, as is the purge stream with flow controller FC-2. The reactor inlet temperature is controlled with temperature controller TC-1. Finally, the level of the flash vessel is controlled with level controller LC-1.

The process exhibits three complexities that make it challenging for soft sensor development. First, the process topology is complex with many unit operations and controllers. Second, the recycle and heat exchangers add complex interactions. Notably, the case study also introduces a cyclical graph topology in the graph representation. Finally, the reaction kinetics together with the flash vessel introduce a time dependency, making this a dynamic problem. Thus, estimating the target of the ammonia concentration in the product flow poses a significant modeling challenge for ML models.

The process was simulated in Aspen Dynamics-based on the steady state flowsheet provided by Araújo and Skogestad with a Python interface. We generate dynamic data by introducing setpoint changes of up to 10% for all controllers in irregular intervals. We introduce one setpoint change at a time. The dynamic process data is recorded with a frequency of 36 s. We create the dataset by considering all measurements of flow, temperature and pressure of all streams. We further take temperature measurements of the reactor, level measurements of the flash and rotational speed measurements of the compressors. As the target, we measure the product concentration of ammonia in the product stream. The stream data as well as the data from unit operations serves as the input to the model, while the product concentration of ammonia at the current time step is the output of the model. Hereby, the past T time steps are considered for the input of the model. We simulate for 120 h, resulting in over 12,000 data points.



(a) Test Mode I: At test time, the operation is similar to the training time



(b) Test Mode II: At test time, the concentration of water in Blend 1 rises, while it remains low in Blend 2.

Fig. 9. Concentration of water in kg/kg in Blend 1 and Blend 2 over time, for both test Mode I and test Mode II.

5. Results and discussions

In this section, we present and compare the results of the topology-aware GNN to established ML soft sensor modeling methods. We compare our model in all case studies to five ML algorithms commonly used in literature and industry: (1) Random Forest Regression (Breiman, 2001), (2) XGBoost (Friedman et al., 2000), (3) Support Vector Regression (SVR) (Boser et al., 1992), (4) Artificial Neural Network (ANN), (5) Transformer (Sitapure and Kwon, 2023a). All models were implemented in python. Random Forest and SVR were developed using scikit-learn (Pedregosa et al., 2011), while XGBoost was developed using the XGBoost library (Chen and Guestrin, 2016). The ANN model as well as the Transformer model were developed with PyTorch (Paszke et al., 2019), while the topology-aware GNN also utilized PyTorch Geometric (Fey and Lenssen, 2019).

For fairness, we tuned the hyperparameter of each model for each case study on the corresponding validation set. We split the datasets for training, validation, and testing using a chronological split (Botache et al., 2023). For the first two case studies, we split the data of normal operation into the training and validation datasets, while the test set consisted of the test mode data. For the ammonia case study, we split the dataset into 70% for training, 15% for validation and 15% for testing. For training and tuning deep learning models (Topology-aware GNN, ANN, Transformer) we utilized a NVIDIA A100 80 GB PCIe GPU, while for the other ML models (XGBoost, Random Forest, SVR) we utilized a 13th Gen Intel Core i7-1365U CPU as well as Intel Xeon E5-6248R 24C 3.0 GHz CPUs provided by DelftBlue (Delft High Performance Computing Centre (DHPC), 2024). The search space of each model as well as the best found hyperparameter can be found in Appendix A.1. The hyperparameter were found using grid search. The results shown in the following three case studies are always on the independent test set of each respective case study. We calculated each models' performance using the coefficient of determination R^2, the root mean squared error (RMSE), and the Mean Average Error (MAE) for complementary insights. We further benchmarked each model's inference speed for a single prediction using a 13th Gen Intel Core i7-1365U CPU.

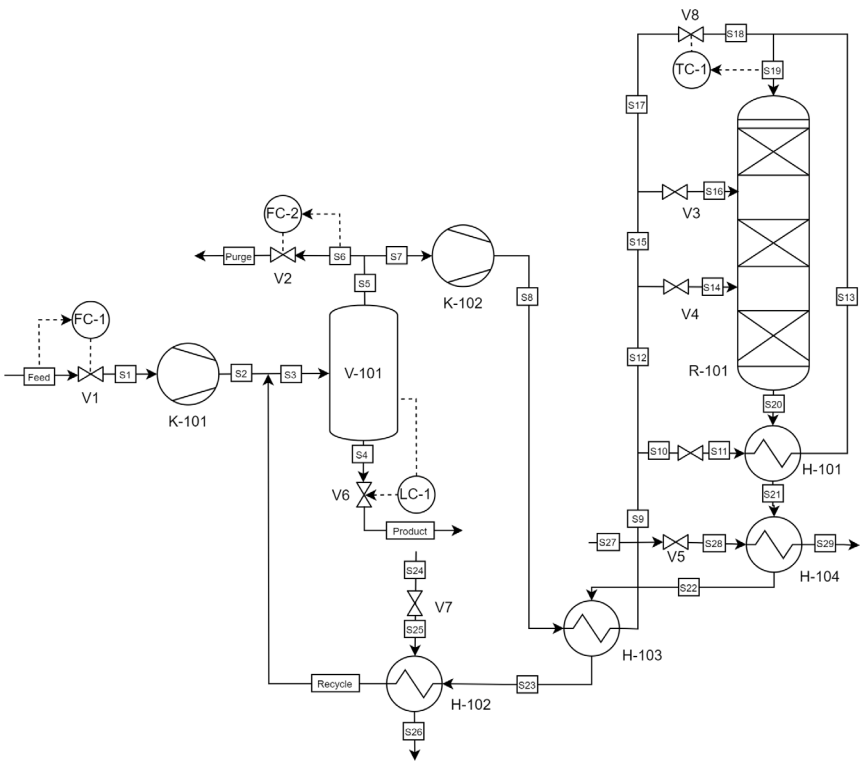


Fig. 10. Ammonia synthesis loop as introduced by Araújo and Skogestad (2008).

Table 3
Comparing our model with various ML models on the pump network case study.

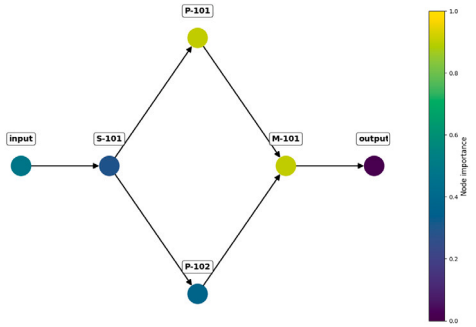
Model	Inference time [ms]	Mode I			Mode II		
		MAE	R^2	RMSE	MAE	R^2	RMSE
ANN	0.17	97.0	0.981	136	3080	−9.46	3240
Random Forest	29.96	823	0.289	842	3330	−11.1	3470
SVR	0.06	965	−0.808	1340	2970	−8.79	3130
XGBoost	0.23	552	0.667	576	3270	−10.7	3420
Transformer	7.80	45.23	0.996	63.0	3094	−9.57	3251
Topology-aware GNN	1.50	**7.32**	**0.999**	**10.0**	**81.5**	**0.991**	**97.2**

5.1. Results on pump network case study

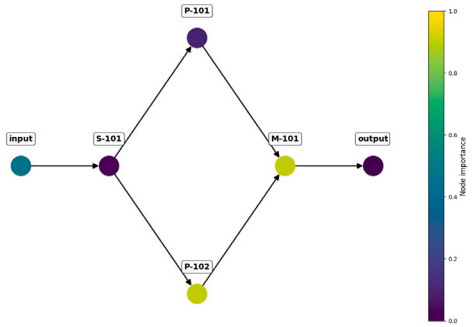
The experimental results for both in-distribution (Mode I) and out-of-distribution (Mode II) scenarios of the pump network case study are presented in Table 3, each shown on the independent test set. Each model was trained with a look back horizon of one time step (36 s). In this case study, all the models were therefore static soft sensors. In Mode I the models show varying degrees of performance. The topology-aware GNN model demonstrates high accuracy across all metrics, achieving an R^2 score of 0.999 and the lowest MAE and RMSE values. Other models, such as ANN, XGBoost, or the Transformer, also perform well in this scenario. Mode II represents a more challenging out-of-distribution scenario, where the flow through P-102 is changed. In this context, the performance of most models degrades significantly, as evidenced by negative R^2 scores and increased error metrics. The topology-aware GNN maintains robust performance even in this out-of-distribution scenario, with an R^2 score of 0.991 and substantially lower MAE and RMSE compared to other models. The inference time between models differs strongly. Most model take one millisecond or less for one prediction, while the Random Forest model takes longest with 29.96 ms.

We hypothesize that the robust performance of the topology-aware GNN in Mode II stems from the flowsheet representation as a graph. With the graph representation, the GNN is made aware that both P-101 and P-102 are pumps through their similar node embeddings. Thus, the GNN model does not differentiate between P-101 and P-102 and transfers the learned behavior from P-101 to P-102, effectively addressing the conditions in Mode II. The transfer is possible because P-102 is operated in Mode II similar to P-101 before. This allows the GNN model to directly transfer the learned behavior between these, making Mode II not an out-of-distribution event.

To further investigate the transfer between the two pumps, we visualize node importance in the topology-aware GNN in Fig. 11. For this, we employ the Graph Neural Network Explainer (GNNExplainer) algorithm to determine the importance of each unit operation (Ying et al., 2019). GNNExplainer identifies the most important subset of graph nodes by optimizing for maximal mutual information with the GNN's output and returns a relative importance score ranging from 0 (node is not important for prediction) to 1 (node is very important for prediction). We apply the GNNExplainer to extract the importance of each node predicting the Mode II validation dataset (normal operation, P-101 varies in operation) and test dataset (out-of-distribution operation, P-102 varies in operation). During normal operation (Fig. 11(a)), the most important nodes are P-101 and M-101 with scores close to 1, indicating these are most critical for estimating the soft sensor target. This aligns with expectations, as P-101 is essential for estimating the

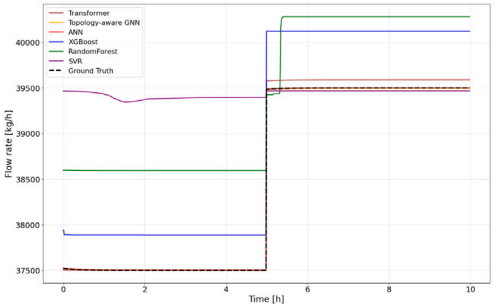


(a) Visualization of the importance of each unit operation in the validation dataset of Mode II, where P-101 is varied in operation and P-102 is kept constant.

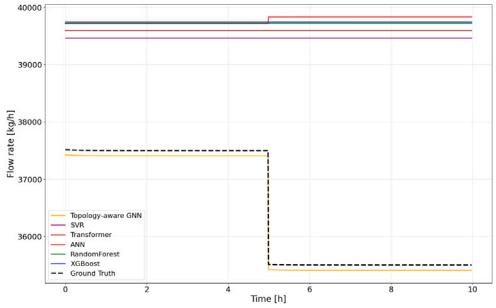


(b) Visualization of the importance of each unit operation in the test dataset of Mode II, where P-101 is kept at a steady flow and P-102 is varied in operation.

Fig. 11. Visualization of the importance of each unit operation in predicting soft sensor targets, visualized with GNNExplainer (Ying et al., 2019).



(a) Visualization of all models predictions on test set for Mode I.



(b) Visualization of all models predictions on test set for Mode II.

Fig. 12. Visualization of model predictions on pump network case study for both test modes.

soft sensor target during normal operation while the flow through P-102 is held constant. Other nodes, such as the splitting unit or P-101 are unimportant, with their respective node importance's below 0.3. During non-normal operation (Fig. 11(b)), the most important nodes are P-102 and M-101, with node importance values close to 1. Other nodes, such as P-101 and S-101 are not important for predicting the soft sensor target in Mode II at test time, with node importance scores below 0.1. This demonstrates that the GNN model detects the operational shift and leverages P-102 instead of P-101 for predicting the soft sensor target in non-normal operation, indicating that transfer learning between the two nodes takes place. Furthermore, one can observe that the output node and the splitter node are not important for the GNN's prediction during either normal or non-normal operation.

The other models (ANN, SVR, XGBoost, Transformer, Random Forest) struggle with the previously unseen situation in Mode II. These standard models are tabular and rely on a fixed input structure. During training, they primarily learn correlations between input sensor data and the soft sensor target. Since the operation of P-102 remains constant in the training set, there is no additional information available about the operation of P-102. These models are not aware that P-101 and P-102 are both pumps and thus behave similarly because their tabular, fixed input treats each input independently. Therefore, when P-102's operation changes at test time, these models fail to accurately predict the soft sensor target as they failed to assign importance to the sensor data around P-102. This reinforces the initial statement from the introduction: flat models have issues adapting to previously unseen operational conditions due to their dependency on fixed input correlations and limited generalization capability.

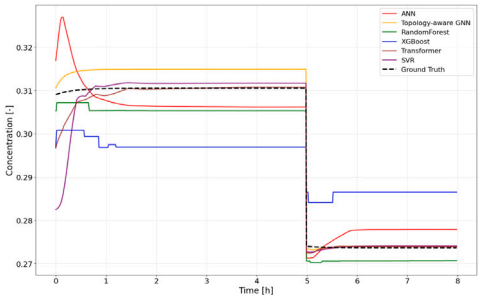
We visualize predictions of all models compared to the ground truth in Fig. 12, showing the differences between Mode I and Mode II. In Mode I, visualized in Fig. 12(a), the seven models' prediction and the ground truth are shown over time. As it can be seen, all models perform well, following the shift in operation at hour 5. In Mode II as visualized in Fig. 12(b) this is not anymore the case. Except for the topology-aware GNN, all models overestimate the total flowrate. This is because the flowrate in P-102 is lowered, lowering the total flowrate. Since these ML models do not take P-102 into account, they overestimate the total flowrate. The topology-aware GNN follows the operation regime well, with a small offset. Further, as the total flowrate changes at hour 5, none of these ML models predict any shift in total flowrate. This is because their predictions rely on P-101, which is not changed in operation at the hour 5. Instead, the flowrate in P-102 is lowered, lowering the total flowrate. The model further correctly recognizes the shift in operation at hour 5.

5.2. Results on blending case study

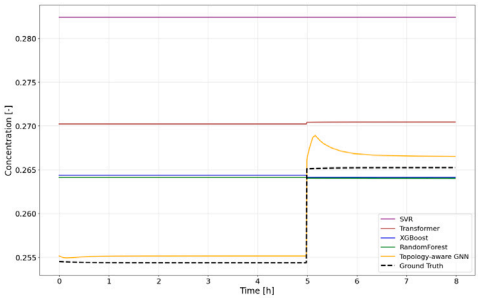
Table 4 presents the performance metrics for various models on the blend network dataset under two scenarios, Mode I and Mode II, each shown on the independent test set. Each model was trained with a look back horizon of one time step (36 s). In this case study, all the models were therefore static soft sensors. In Mode I, where the data maintain its correlation structure, most models demonstrate good performance. The XGBoost model achieves the best results in this scenario, with the highest R^2 score of 0.904 and the lowest MAE and RMSE values. Other models, such as Random Forest and the topology-aware GNN, also perform well, with R^2 scores above 0.88. Mode II represents a more challenging scenario as the correlation structure is disrupted. This change significantly impacts the performance of most models, as

Table 4
Comparing our model with various ML models on the blend network case study.

Model	Inference time [ms]	Mode I			Mode II		
		MAE	R^2	RMSE	MAE	R^2	RMSE
ANN	0.18	**0.0299**	**0.997**	**0.0436**	300	−2.17e+06	368
Random Forest	17.60	0.200	0.941	0.207	0.315	−1.30	0.379
SVR	0.23	0.0811	0.937	0.214	1.23	−24.3	1.26
XGBoost	0.17	0.616	0.465	0.620	0.348	−1.00	0.354
Topology-aware GNN	1.79	0.134	0.963	0.163	**0.0518**	**0.956**	**0.0525**
Transformer	2.08	0.0364	0.989	0.0893	0.566	−5.08	0.617



(a) Visualization of all models predictions on test set for Mode I.



(b) Visualization of all models predictions on test set for Mode II. The ANN model's results for Mode II were not plotted because its large prediction errors would have compressed the visualization of the other models' performances.

Fig. 13. Visualization of model predictions on blend network case study for both test modes.

evidenced by negative R^2 scores and increased error metrics. Only the topology-aware GNN maintains robust performance even in this altered scenario, achieving an R^2 score of 0.956 and substantially lower MAE and RMSE compared to other models.

We hypothesize the GNN recognizes the lack of causation between the concentration of water in Blend 2 due to its topology-awareness during its training phase and therefore also does not consider it at test time. The GNN model's performance in Mode II further benefits from the consistent ethanol concentration in Blend 1, which is similar to previous levels. This consistency ensures that Mode II is treated as an in-distribution event, allowing the model to perform effectively. The information flow in the GNN is bound by the process topology, enforcing a correlation structure-based on the underlying process. The flowsheet topology is communicated through the directed graph. Since the direction of the graph also determines the direction of the information flow in the GNN in the message passing phase, the topology effectively restricts the GNN to consider only sensor information upstream of Blend 1 and thus ignore Blend 2. We further investigate the importance of the flow of information in Ablation 6.2. Other ML models do not have this access to the flowsheet information. During training time, they therefore learn that the concentration in Blend 2 is a good predictor for the concentration in Blend 1 because they are highly correlated. Since this relationship is broken in Mode II, the ML models' accuracy is also low as this correlation no longer holds true. Similar to the blend case study, most models require around one millisecond for inference. A notable outlier is here again the random forest model with 17.60 ms.

To further support this explanation, we extracted the feature importance from the second-best performing model, XGBoost. The five most important features (accounting for 97.55%) were the pressure in Blend 2 (27.56%), the temperature of the ethanol stream (21.51%), the temperature of Blend 2 (17.69%), the concentration of ethanol in Blend 2 (17.00%) and the temperature of Blend 1. Notably, among the five most important features only one is related to Blend 1, and three are related to Blend 2. This confirms that XGBoost learns correlations between features in Blend 1 and Blend 2 rather than the causal relationship for Blend 1.

We further plot the predictions on the test set for the two modes with all models in Fig. 13. In Mode I shown in Fig. 13(a), all models follow the concentration profile well. In Mode II as shown in Fig. 13(b) this is no longer the case. All models except for the topology-aware GNN are overestimating the concentration in Blend 1. This is because the concentration in Blend 2 is held constant at test time, while the concentration in Blend 1 is lowered. As the ML models rely on Blend 2 to predict Blend 1, they are overestimating the concentration. The topology aware GNN predicts the concentration correctly, with a small offset. Further, at hour 5, the operation is shifted and the concentration in Blend 1 increased. As the concentration in Blend 2 is held almost constant, the other ML do not detect this shift and accordingly do not change their predicted concentration.

Case study 1 and 2 can also be modeled without using the topology-aware approach. For example, utilizing the parallel structure of the two pumps, a hybrid ML model (Schweidtmann et al., 2024; Bradley et al., 2022) could be developed that builds on mass balances. Similarly, excluding the concentration of Blend 2 would be a possible solution to avoid the spurious relationship between the concentration measurements in Blend 1 and Blend 2. In real plants however, such situations may not be as obvious due to a large number of unit operations, complex piping networks and an overwhelming number of sensors. The topology-aware approach learns to utilize the plant topology without the need to previously identify such pitfalls and without any further manual modeling effort required.

5.3. Results on ammonia synthesis simulation

We tested the topology-aware approach on the ammonia synthesis process for both overall performance as well as performance in a low data regime on the independent test set. The results on the test set can be found in Table 5. Each model shown was trained with a look back horizon of 20 time steps (12 min). In this case study, all the models were therefore dynamic soft sensors, not only taking the last time step into consideration but also previous measurements. As it can be seen, our topology-aware GNN outperforms other models significantly

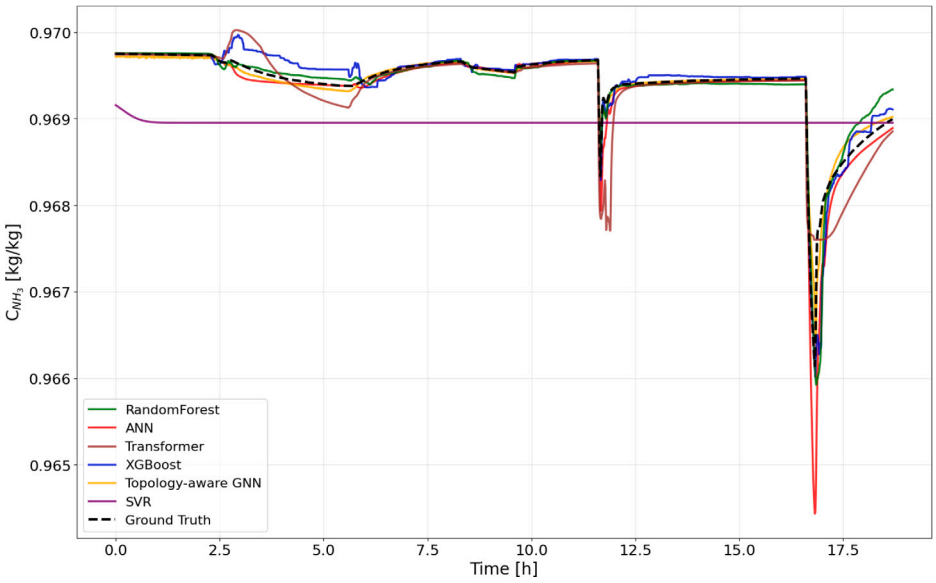


Fig. 14. Prediction of ML models on ammonia case study test set over time.

Table 5
Performance comparison of topology-aware GNN (ours) to other common regression models. Best performing models per metric highlighted in bold.

Model	Inference time [ms]	MAE	R^2	RMSE
ANN	0.31	0.186	0.855	0.317
RandomForest	23.17	0.185	0.836	0.322
SVR	1.01	0.856	−0.492	0.972
XGBoost	1.12	0.162	0.856	0.302
Topology-aware GNN	26.72	**0.121**	**0.944**	**0.187**
Transformer	4.61	0.217	0.677	0.451

with an R^2 of 0.944 compared to an R^2 of 0.856 for the second best model, XGBoost. The results demonstrate that including the topology into the prediction process helps in modeling the process. In Fig. 14 we visualize all ML models' prediction of the ammonia concentration over time on the test set of the ammonia case study. Most ML models predict the ammonia concentration precisely, in accordance with the performance metrics. One exception is the SVR model, which does not perform well. The inference speed indicates that all models take longer to make a prediction than in previous case studies, due to the large number of features in the predictions, with most models now requiring more than one millisecond. Especially the topology-aware GNN now requires 26.72 ms for this much bigger case study.

We further test how data efficient our modeling approach is on the ammonia dataset. Towards this end, we train the GNN model on fractions of the original training dataset. We remove a portion of the training data from the time series by excluding a block of the training and validation observations, ensuring a true reduction in the training dataset. We then test the trained model on the test dataset. For comparison, we consider our second best model, XGBoost. We train each model five times to account for training noise. The results can be found in Fig. 15, where we compare the two models with their MAE. As expected, the performance of both models deteriorates when being trained on less data, with the MAE rising to 3.17 for the topology-aware GNN and 3.22 for XGBoost on 10% of the original data. Another notable trend is that in a low data regime, the GNN model seems to predict significantly more accurate, with a 14.8% and 15.7% reduction in MAE at 30% and 20% of the whole dataset. At 10% of the original dataset, this effect fades. Overall, the GNN model is more accurate under all considered data fractions. We attribute this to the topology-awareness, making the GNN require less data.

6. Ablation studies

In this section, we further investigate the internal modeling mechanisms of the topology-aware GNN approach through ablation studies. In the first part, we alter the node representation defined by the topology and retrain on the pump network case study to investigate if the model is still able to transfer behavior between the two pumps. In the second part, we study the influence of the directionality of the graphs onto the blend case study. In the third, we further alter the pump case study to break its inherent symmetry.

6.1. Influence of node representation in the process topology

We previously hypothesized that node representations aid the GNN model to transfer learned dynamics from P-101 to P-102 due to the node information given in the process topology. We argue that this is due to the representation chosen. To test this, we consider three ablations to the original representation.

In the original case, the node attributes x_{P-101} for P-101 and x_{P-102} for P-102 are similar with $\mathbf{x}_{P\text{-}101} = \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0 & S_{P-101}(t)\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{x}_{P\text{-}102} = \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0 & S_{P-102}(t)\end{bmatrix}^{\mathrm{T}}$, whereby $S_{P-101}(t)$ and $S_{P-102}(t)$ denote the sensor measurements at P-101 and P-102 respectively, and the type of unit operation is one hot encoded. Each node attribute x is of size seven due to four types of nodes (Inlet/Outlet, Mixer, Splitter, Pump) and three types of measurements (Flow, Pressure, Temperature).

In the first ablation, we remove node type information. In this ablation, the topology graph thus only contains information about the interconnectivity between nodes and the node attributes only contain sensor measurements. The encoded node attributes x_{P-101} for P-101 and x_{P-102} for P-102 then become $\mathbf{x}_{P\text{-}101} = \begin{bmatrix}0 & 0 & SP-101(t)\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{x}_{P\text{-}102} = \begin{bmatrix}0 & 0 & SP-102(t)\end{bmatrix}^{\mathrm{T}}$. The overall embedding vector size is reduced, as the one-hot encoding is not applied. However, both P-101 and P-102 can still have identical embeddings.

In the second ablation. we heterogenize the measurement type, meaning that each measurement gets a unique field. In this ablation, each sensor measurement from any equipment gets its own field, meaning that temperatures or flows of different units/streams no longer share channels. The encoded node attributes x_{P-101} for P-101 and x_{P-102} for P-102 then become

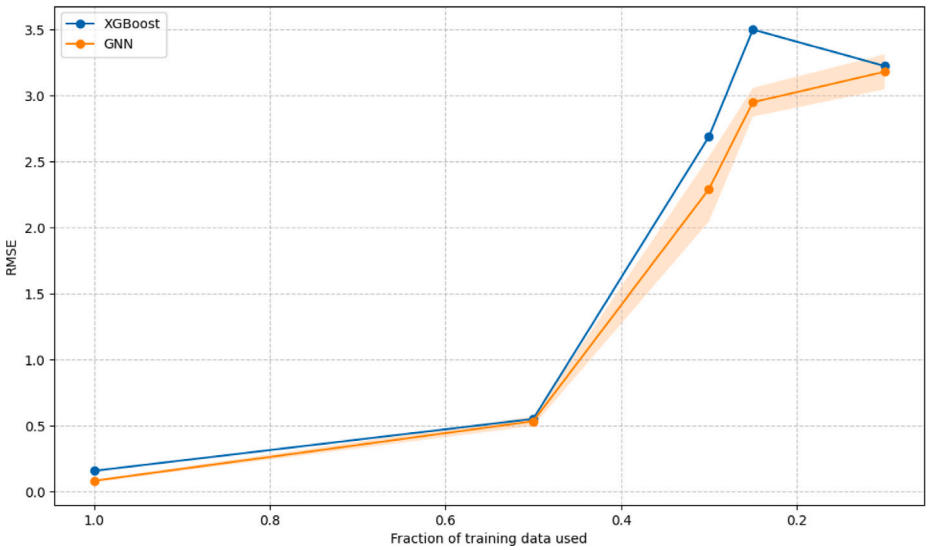


Fig. 15. Comparison of topology-aware GNN and XGBoost when trained on a fraction of the dataset. A lower mean average error is better.

Table 6
Comparison of original GNN and ablation variants on the pump network case study (rounded to three significant digits).

#	Ablation	Mode I			Mode II		
		MAE	R^2	RMSE	MAE	R^2	RMSE
Original	Original Topology-aware GNN	7.32	0.999	10.0	81.5	0.991	97.2
1	No unit type information	2.07	0.999	2.14	413	0.830	413
2	Heterogenized measurement type	3.43	1.00	3.61	3070	−9.43	3230
3	No unit type + heterogenized measurement	16.1	1.00	17.1	5090	−27.1	5300

$\mathbf{x}_{P\text{-}101} = \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & S_{P-101}(t) & 0\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{x}_{P\text{-}102} = \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & S_{P-102}(t)\end{bmatrix}^{\mathrm{T}}$. In this representation, the encoded node attribute vector becomes larger, as each sensor in the process has a unique channel in the embedding. Here, $S_{P-101}(t)$ and $S_{P-102}(t)$ do not share the same channel any longer. However, in this representation, P-101 and P-102 still share the one-hot encoded unit type information.

In the third ablation, we remove both unit type information and heterogenize the measurement type. In this ablation, each sensor measurement has a unique field and there are no unit type information. Accordingly, the encoded node attributes for P-101 and P-102 are $\mathbf{x}_{P\text{-}101} = \begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & S_{P-102}(t) & 0\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{x}_{P\text{-}102} = \begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0 & S_{P-102}(t)\end{bmatrix}^{\mathrm{T}}$. In this way, each pump has a completely unique encoding.

In Table 6, we show the results of these three ablations compared to the original GNN. For Mode I, the GNN model still performs similarly well under the applied ablations with R^2 values larger than 0.999. Thus, neither unit type information nor the encoding of the measurement type are relevant to predict the soft sensor target here. This is also in accordance with the very good performance of the other ML models that do not leverage any topology information on the pump network case study in Mode I. In Mode II, the GNN model does not perform as well anymore with the ablated data representations. When removing the unit type information in Ablation 1, the model performance deteriorates, with a substantial increase in error metrics and a drop in R^2 to 0.83. Removing the encoding of the measurement type in Ablation 2 leads to an even stronger degradation, resulting in large prediction errors and a negative R^2 value. The combination of both ablations, Ablation 3, produces the worst performance, with extreme error magnitudes and R^2 far below zero, indicating that the model fails to capture the underlying relationships in this setting, similar to other ML models. We hypothesize that in Ablation 1 the model still performs well because the underlying encoding of the two pumps is still identical, thus allowing transfer learning, albeit less due to the missing unit type information. When heterogenizing the sensor measurements in the last two cases, the two pumps no longer share any common embedding. Thus, transfer learning is not possible any longer.

6.2. Influence of process topology graph on information flow

In order to test the importance of controlled information flow in the GNN, we investigate the influence of the directionality and location in the process topology graph in the blend case study. For this, we introduce two ablations to the GNN model in the blend case study. First, we introduce undirected message passing. Here, the flow of information during the message passing is no longer unidirectional but can also proceed against the stream's direction. Second, we utilize graph-based aggregation instead of node-based aggregation after the message passing step. This way, information that is far away from the target node is still incorporated in the flowsheet fingerprint and afterwards in the prediction step. This may enable information exchanged across nodes that are uncorrelated and spatially distant.

In Table 7 are the results of both ablations individually as well as the original GNN. For Mode I, the original model as well as the ablations perform well, with each R^2 over 0.96. Out of the three, the original model does not perform best. This may indicate that the control of the information flow adversely affects modeling since spurious correlations between predictors and target cannot be leveraged. These spurious correlations provide predictive power in Mode I, where Blend 1 and Blend 2 are highly correlated. In Mode II, the original model then outperforms both ablations with an R^2 of 0.956 compared to an R^2 of 0.900 for the undirected message passing in Ablation 1 and an R^2 of −647 for the graph-level aggregation in Ablation 2. Not including the directionality

Table 7
Comparison of the original GNN and its ablation variants on the blend case study.

#	Ablation	Mode I			Mode II		
		MAE	R^2	RMSE	MAE	R^2	RMSE
Original	Original GNN	0.134	0.963	0.163	0.0518	0.956	0.0525
1	Undirected message passing	0.157	0.950	0.190	0.0552	0.901	0.0789
2	Graph-aggregation	0.0452	0.997	0.0492	6.17	−647	6.37

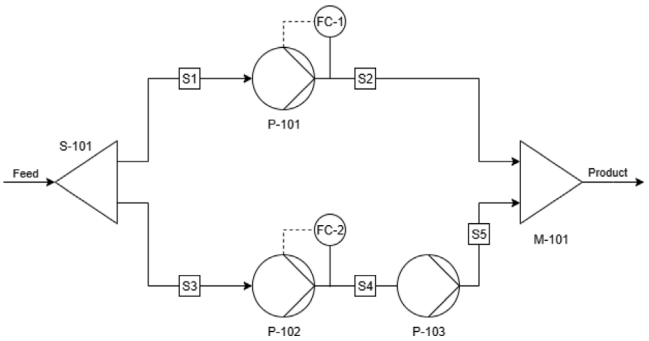


Fig. 16. Asymmetric pump flowsheet with P-103 added.

Table 8
Performance comparison of GNN and XGBoost on asymmetric pump case study.

Model	Mode I			Mode II		
	MAE	R^2	RMSE	MAE	R^2	RMSE
Topology-aware GNN	45.2	0.997	58.1	81.3	0.987	113
XGBoost	427	0.782	466	1000	−0.134	1060

as in Ablation 1 does negatively affect modeling performance, but nevertheless the model still captures the break in correlations between the two concentrations well. One possible explanation may be that the information exchange between the two blends is limited due to their distance in the graph. In the blend process topology, the shortest path between the two outlet nodes consists of four edges. GNNs can struggle to exchange information over such long ranges (Jain et al., 2021). In contrast, when all node information is aggregated as it is carried out in the Ablation 2, the model can fully leverage the correlation between two concentrations, leading to complete failure in Mode II. Thus, this ablation study highlights that the topology aware approach can help avoid spurious relationships compared to other ML methods through a controlled flow of information.

6.3. Influence of symmetry in pump network case study

To investigate the influence of symmetry on the predictive capabilities of the GNN model in the pump case study, we alter the pump network. As shown in Fig. 16, in this ablation study a third pump P-103 in row with P-102 is added. P-102 and P-103 increase the pressure equally. The operating profile is kept identical to before with two distinct modes. In Mode I the flow is varied through P-101 at training and test time, while P-102 and P-103 are kept constant. In Mode II, P-102 is varied in operation at test time, which alters the flow through P-102 and P-103, while P-101 is kept constant. We train the GNN model on this ablated case study, as well as XGBoost as a comparison.

The results of the ablation can be found in Table 8, indicating that breaking the symmetry does not adversely affect the predictive performance of the GNN model. In Mode I, the GNN model outperforms the XGBoost model at test time, wit h an R^2 of 0.996 vs 0.782 for XGBoost. In the more challenging Mode II with asymmetric pumps, the GNN model still performs robustly, with an R^2 of 0.987. Although its performance slightly deteriorates, as seen by a 94.1% increase in RMSE from 58.1 to 113, it is a stark contrast to the XGBoost model, whose performance collapses entirely, yielding a negative R^2 of −0.134. Overall, the asymmetric ablations demonstrates that GNN model learns symmetry independent to predict the pump behavior. Even with P-102 and P-103 in a row, the model still correctly predicts the total flow through the network.

7. Conclusion

We present a novel approach for ML-based modeling of soft sensors. We encode the process topology and sensor data into a unified context via graph representations, which we then process with a GNN. We have shown that our approach has inherent advantages for process modeling over pure ML models in certain situations and demonstrated those advantages with a few case studies. Its graph structure can enable transfer learning between units, the topology can aid in enforcing causality and the process context reduces data requirements and enhances performance. All of these properties are of great use when developing ML-based process models. We further argue that the process topology is available and known for almost any industrial process. Thus, the added modeling effort compared to pure ML methods is marginal.

We foresee several promising directions of future work towards digital twins. The presented method could be applied to real process data from industrial plants with complex processing where the plant topology will aid in modeling. Another promising avenue would be to incorporate the developed GNNs into a model predictive control formulation. In this context, control information could be incorporated into the graph-based process topology representation, e.g., with controllers as nodes and control signals as edges. We aim to train GNNs beyond a single process by building different graphs for different process topologies. This would enable transferring ML models to similar processes or train ML models on data from multiple, topologically different plants.

CRediT authorship contribution statement

Maximilian F. Theisen: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Gabrie M.H. Meesters:** Writing – review & editing, Validation, Supervision, Funding acquisition, Formal analysis, Conceptualization. **Artur M. Schweidtmann:** Writing – review & editing, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

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.

Acknowledgments

This research is part of the Process & Product Technology Institute (Pro2Tech) research incubator, for which we express sincere gratitude.

Table 9
Hyperparameter tuning for topology-aware GNN.

Hyperparameter	Values	Pump network	Blend network	Ammonia synthesis
Batch Size	[10, 50, 100, 400]	10	50	100
Learning Rate	[5e−4, 1e−4, 5e−3, 1e−3, 5e-2]	0.001	0.0005	0.0005
Hidden Dimension Size	[8, 32, 128]	8	32	32
Number of Layers	[1, 4, 6, 8]	1	6	4
Pooling Type	[max, mean, sum]	max	max	max
Dropout Probability	[0.1, 0.3, 0.5]	0.1	0.1	0.1
Node or graph embedding	[graph, node]	graph	node	graph

Table 10
Hyperparameter tuning for support vector regression.

Hyperparameter	Values	Pump network	Blend network	Ammonia synthesis
C (Penalty Parameter)	[0.1, 1, 10]	10	0.1	10
Epsilon (Loss Function)	[0.01, 0.1, 1]	0.01	0.01	0.01
Gamma (Kernel Coefficient)	[0.1, 1, 10, scale, auto]	auto	auto	auto
Kernel Type	[poly, rbf, sigmoid]	rbf	rbf	rbf

Table 11
Hyperparameter tuning for random forest regression.

Hyperparameter	Values	Pump network	Blend network	Ammonia synthesis
Number of Estimators	[500, 750, 1000]	1000	500	750
Maximum Depth	[null, 3, 9, 15]	15	9	9
Maximum Features	[1.0, log2, sqrt]	sqrt	log2	log2
Minimum Samples Split	[2, 4, 6]	6	2	4

Table 12
Hyperparameter tuning for artificial neural network.

Hyperparameter	Values	Pump network	Blend network	Ammonia synthesis
Batch Size	[10, 50, 100, 400]	10	50	100
Learning Rate	[5e−4, 1e−4, 5e−3, 1e−3, 5e-2]	0.05	0.0005	0.0005
Weight Decay	[0, 1e−6, 1e−5, 1e-4]	1e−4	1e−4	0
Hidden Size	[8, 32, 128, 256]	8	128	32

Table 13
Hyperparameter tuning for XGBoost.

Hyperparameter	Values	Pump network	Blend network	Ammonia synthesis
Learning Rate	[0.01, 0.05, 0.1, 0.2]	0.01	0.01	0.2
Maximum Depth	[3, 7, 12]	3	3	7
Subsample	[0.5, 0.75, 1.0]	1.0	0.5	0.5
Number of Estimators	[50, 100, 200, 300, 500]	500	100	500
Column Sample by Tree	[0.5, 0.7, 0.9]	0.9	0.7	0.5
Gamma	[0.0, 0.1, 0.3]	0.1	0.0	0.3

Table 14
Hyperparameter tuning for transformer.

Hyperparameter	Values	Pump network	Blend network	Ammonia synthesis
Batch Size	[32, 64, 128]	32	32	32
Learning Rate	$[10^{-5}, 5\times10^{-5}, 10^{-4}, 5\times10^{-4}]$	10^{-4}	10^{-5}	5×10^{-4}
Hidden Size	[64, 128, 256]	128	256	64
Number of Heads	[4, 8, 16]	4	4	8
Number of Layers	[3, 6, 9]	6	3	3

Appendix. Supplementary information

A.1. Hyperparameter tuning results

See Tables 9–14.

Data availability

The data that has been used is confidential.

References

Allen, Louis, Cordiner, Joan, 2024. Towards sustainable WWTP operations: Forecasting energy consumption with explainable disentangled graph convolutional networks. In: Manenti, Flavio, Reklaitis, Gintaras V. (Eds.), Proceedings of the 34th European Symposium on Computer Aided Process Engineering / 15th International Symposium on Process Systems Engineering (ESCAPE34/PSE24). In: Computer Aided Chemical Engineering, vol. 50, Elsevier, pp. 1567–1572.

Araújo, Antonio, Skogestad, Sigurd, 2008. Control structure design for the ammonia synthesis process. Comput. Chem. Eng. 32 (12), 2920–2932.

Arce Munoz, Samuel, Hedengren, John D., 2025. Transfer learning for thickener control. Processes 13 (1), 223.

Arce Munoz, Samuel, Pershing, Jonathan, Hedengren, John D., 2024. Physics-informed transfer learning for process control applications. Ind. Eng. Chem. Res. 63 (49), 21432–21443.

Bai, Yiming, Zhao, Jinsong, 2023a. A novel transformer-based multi-variable multi-step prediction method for chemical process fault prognosis. Process. Saf. Environ. Prot. 169, 937–947.

Bai, Yiming, Zhao, Jinsong, 2023b. A process data prediction method for chemical process based on the frozen pretrained transformer model. In: 33rd European Symposium on Computer Aided Process Engineering. Elsevier, pp. 1717–1723.

Balhorn, Lukas Schulze, Degens, Kevin, Schweidtmann, Artur M., 2024. Graph-to-SFILES: Control structure prediction from process topologies using generative artificial intelligence.

Battaglia, Peter W, Hamrick, Jessica B, Bapst, Victor, Sanchez-Gonzalez, Alvaro, Zambaldi, Vinicius, Malinowski, Mateusz, Tacchetti, Andrea, Raposo, David, Santoro, Adam, Faulkner, Ryan, et al., 2018. Relational inductive biases, deep learning, and graph networks.

Bortz, Michael, Dadhe, Kai, Engell, Sebastian, Gepert, Vanessa, Kockmann, Norbert, Müller-Pfefferkorn, Ralph, Schindler, Thorsten, Urbas, Leon, 2023. AI in process industries – current status and future prospects. Chem. Ing. Tech. 95 (7), 975–988.

Boser, Bernhard E., Guyon, Isabelle M., Vapnik, Vladimir N., 1992. A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory. COLT '92, Association for Computing Machinery, New York, NY, USA, pp. 144–152.

Botache, Diego, Dingel, Kristina, Huhnstock, Rico, Ehresmann, Arno, Sick, Bernhard, 2023. Unraveling the complexity of splitting sequential data: Tackling challenges in video and time series analysis.

Bradley, William, Kim, Jinhyeun, Kilwein, Zachary, Blakely, Logan, Eydenberg, Michael, Jalvin, Jordan, Laird, Carl, Boukouvala, Fani, 2022. Perspectives on the integration between first-principles and data-driven modeling. Comput. Chem. Eng. 166, 107898.

Breiman, Leo, 2001. Random forests. Mach. Learn. 45 (1), 5–32.

Cao, Liang, Su, Jianping, Wang, Yixiu, Cao, Yankai, Siang, Lim C., Li, Jin, Saddler, Jack Nicholas, Gopaluni, Bhushan, 2022. Causal discovery based on observational data and process knowledge in industrial processes. Ind. Eng. Chem. Res. 61 (38), 14272–14283.

Chen, Tianqi, Guestrin, Carlos, 2016. XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '16, ACM.

Cheng, Qiu, Chunhong, Zhan, Qianglin, Li, 2023. Development and application of random forest regression soft sensor model for treating domestic wastewater in a sequencing batch reactor. Sci. Rep. 13 (1).

Ching, Phoebe Mae Lim, Zou, Xu, Wu, Di, So, Richard Hau Yue, Chen, Guanghao, 2022. Development of a wide-range soft sensor for predicting wastewater BOD5 using an extreme gradient boosting (XGBoost) machine. Environ. Res. 210, 112953.

Coley, Connor W., Jin, Wengong, Rogers, Luke, Jamison, Timothy F., Jaakkola, Tommi S., Green, William H., Barzilay, Regina, Jensen, Klavs F., 2019. A graph-convolutional neural network model for the prediction of chemical reactivity. Chem. Sci. 10 (2), 370–377.

Curreri, Francesco, Patanè, Luca, Xibilia, Maria Gabriella, 2021. Soft sensor transferability: A survey. Appl. Sci. 11 (16), 7710.

d'Anterroches, Loïc, 2006. Process Flow Sheet Generation Design Through a Group Contribution Approach. Technical University of Denmark, Kgs. Lyngby.

Daoutidis, Prodromos, Lee, Jay H., Rangarajan, Srinivas, Chiang, Leo, Gopaluni, Bhushan, Schweidtmann, Artur M., Harjunkoski, Iiro, Mercangöz, Mehmet, Mesbah, Ali, Boukouvala, Fani, Lima, Fernando V., del Rio Chanona, Antonio, Georgakis, Christos, 2024. Machine learning in process systems engineering: Challenges and opportunities. Comput. Chem. Eng. 181, 108523.

Delft High Performance Computing Centre (DHPC), 2024. DelftBlue supercomputer (phase 2). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2.

Dey, Rahul, Salem, Fathi M., 2017. Gate-variants of Gated Recurrent Unit (GRU) neural networks. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS). pp. 1597–1600, ISSN: 1558-3899.

Dosovitskiy, Alexey, Beyer, Lucas, Kolesnikov, Alexander, Weissenborn, Dirk, Zhai, Xiaohua, Unterthiner, Thomas, Dehghani, Mostafa, Minderer, Matthias, Heigold, Georg, Gelly, Sylvain, Uszkoreit, Jakob, Houlsby, Neil, 2021. An image is worth 16x16 words: Transformers for image recognition at scale.

Fey, Matthias, Lenssen, Jan E., 2019. Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds.

Fortuna, Luigi, Graziani, Salvatore, Rizzo, Alessandro, Xibilia, Maria G (Eds.), 2007. Soft sensors for monitoring and control of industrial processes. In: SpringerLink, Springer London, London.

Friedman, Jerome, Hastie, Trevor, Tibshirani, Robert, 2000. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann. Statist. 28 (2).

Gao, Qinghe, Schweidtmann, Artur M., 2024. Deep reinforcement learning for process design: Review and perspective. Curr. Opin. Chem. Eng. 44, 101012.

Gilmer, Justin, Schoenholz, Samuel S., Riley, Patrick F., Vinyals, Oriol, Dahl, George E., 2017. Neural message passing for quantum chemistry. In: Precup, Doina, Teh, Yee Whye (Eds.), Proceedings of the 34th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 70, PMLR, Sydney, Australia, pp. 1263–1272, URL https://proceedings.mlr.press/v70/gilmer17a.html.

Goldstein, Dominik P., Alimin, Achmad Anggawirya, Schulze Balhorn, Lukas, Schweidtmann, Artur M, 2025. pyDEXPI:A Python framework for piping and instrumentation diagrams using the DEXPI information model. In: Proceedings of the 35th European Symposium on Computer Aided Process Engineering. (ESCAPE35), Ghent, Belgium.

Grimstad, Bjarne, Løvland, Kristian, Imsland, Lars S., 2023. Multi-unit soft sensing permits few-shot learning. ArXiv.

Hamilton, William L., 2020a. Graph representation learning. Synth. Lect. Artif. Intell. Mach. Learn. 14 (3), 1–159.

Hamilton, William L., 2020b. Graph representation learning. In: Synthesis Lectures on Artificial Intelligence and Machine Learning, (3), Morgan & Claypool Publishers, pp. 1–159.

Hochreiter, Sepp, Schmidhuber, Jürgen, 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.

Hu, Wenkai, Shah, Sirish L., Chen, Tongwen, 2018. Framework for a smart data analytics platform towards process monitoring and alarm management. Comput. Chem. Eng. 114, 225–244.

Huang, Yu, Zhang, Chao, Yella, Jaswanth, Petrov, Sergei, Qian, Xiaoye, Tang, Yufei, Zhu, Xingquan, Bom, Sthitie, 2021. GraSSNet: Graph soft sensing neural networks. In: 2021 IEEE International Conference on Big Data (Big Data). IEEE.

Jain, Paras, Wu, Zhanghao, Wright, Matthew A., Mirhoseini, Azalia, Gonzalez, Joseph E., Stoica, Ion, 2021. Representing long-range context for graph neural networks with global attention. In: Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J. Wortman (Eds.), Advances in Neural Information Processing Systems.

Jiang, Yuanyuan, Yang, Zongwei, Guo, Jiali, Li, Hongzhen, Liu, Yijing, Guo, Yanzhi, Li, Menglong, Pu, Xuemei, 2021a. Coupling complementary strategy to flexible graph neural network for quick discovery of coformer in diverse co-crystal materials. Nat. Commun. 12 (1).

Jiang, Yuchen, Yin, Shen, Dong, Jingwei, Kaynak, Okyay, 2021b. A review on soft sensors for monitoring, control, and optimization of industrial processes. IEEE Sensors J. 21, 12868–12881.

Jiang, Yuchen, Yin, Shen, Dong, Jingwei, Kaynak, Okyay, 2021c. A review on soft sensors for monitoring, control, and optimization of industrial processes. IEEE Sensors J. 21 (11), 12868–12881.

Kadlec, Petr, Gabrys, Bogdan, Strandt, Sibylle, 2009. Data-driven soft sensors in the process industry. Comput. Chem. Eng. 33 (4), 795–814.

Kaneko, Hiromasa, Funatsu, Kimito, 2014. Adaptive soft sensor based on online support vector regression and Bayesian ensemble learning for various states in chemical plants. Chemometr. Intell. Lab. Syst. 137, 57–66.

Kipf, Thomas N., Welling, Max, 2017. Semi-supervised classification with graph convolutional networks. In: Proceedings of the 5th International Conference on Learning Representations. ICLR.

Klaeger, Tilman, Gottschall, Sebastian, Oehm, Lukas, 2021. Data science on industrial data—Today's challenges in brown field applications. Challenges 12 (1), 2.

Lastrucci, Giacomo, Theisen, Maximilian F., Schweidtmann, Artur M., 2024. Physics-informed neural networks and time-series transformer for modeling of chemical reactors. In: 34th European Symposium on Computer Aided Process Engineering / 15th International Symposium on Process Systems Engineering. Elsevier, pp. 571–576.

Le, Phong, Zuidema, Willem, 2016. Quantifying the vanishing gradient and long distance dependency problem in recursive neural networks and recursive LSTMs. In: Proceedings of the 1st Workshop on Representation Learning for NLP. Association for Computational Linguistics, Berlin, Germany, pp. 87–93.

Li, Jianping, 2024. Learning hybrid extraction and distillation using phenomena-based string representation. In: Proceedings of the 10th International Conference on Foundations of Computer-Aided Process Design FOCAPD 2024. In: FOCAPD 2024, vol. 3, PSE Press, pp. 300–307.

Lin, Xiaoyong, Li, Zihui, Han, Yongming, Chen, Zhiwei, Geng, Zhiqiang, 2024. Novel spatiotemporal graph attention model for production prediction and energy structure optimization of propylene production processes. Comput. Chem. Eng. 181, 108507.

Lin, Bao, Recke, Bodil, Knudsen, Jørgen K.H., Jørgensen, Sten Bay, 2007. A systematic approach for soft sensor development. Comput. Chem. Eng. 31 (5–6), 419–425.

Liu, Yang, Lapata, Mirella, 2019. Text Summarization with Pretrained Encoders. arXiv:1908.08345 [cs].

Liu, X., Yang, N., Jiang, Y., et al., 2020. A parallel computing-based deep attention model for named entity recognition. J. Supercomput. 76, 814–830.

Lu, Jie, Liu, Anjin, Dong, Fan, Gu, Feng, Gama, Joao, Zhang, Guangquan, 2020. Learning under concept drift: A review. IEEE Trans. Knowl. Data Eng. 31 No. 12 (2018) 2346-2363 1–1.

Luttmann, Reiner, Bracewell, Daniel G., Cornelissen, Gesine, Gernaey, Krist V., Glassey, Jarka, Hass, Volker C., Kaiser, Christian, Preusse, Christian, Striedner, Gerald, Mandenius, Carl-Fredrik, 2012. Soft sensors in bioprocessing: A status report and recommendations. Biotechnol. J. 7 (8), 1040–1048.

Mann, Vipul, Sales-Cruz, Mauricio, Gani, Rafiqul, Venkatasubramanian, Venkat, 2024. eSFILES: Intelligent process flowsheet synthesis using process knowledge, symbolic AI, and machine learning. Comput. Chem. Eng. 181, 108505.

Marquardt, Wolfgang, Morbach, Jan, Wiesner, Andreas, Yang, Aidong, 2010. OntoCAPE: A Re-Usable Ontology for Chemical Process Engineering. Springer Berlin Heidelberg.

Niresi, Keivan Faghih, Bissig, Hugo, Baumann, Henri, Fink, Olga, 2024. Physics-enhanced graph neural networks for soft sensing in industrial internet of things. IEEE Internet Things J. 1–1.

Oeing, Jonas, Brandt, Kevin, Wiedau, Michael, Tolksdorf, Gregor, Welscher, Wolfgang, Kockmann, Norbert, 2023. Graph learning in machine-readable plant topology data. Chem. Ing. Tech. 95 (7), 1049–1060.

Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, Killeen, Trevor, Lin, Zeming, Gimelshein, Natalia, Antiga, Luca, Desmaison, Alban, Kopf, Andreas, Yang, Edward, DeVito, Zachary, Raison, Martin, Tejani, Alykhan, Chilamkurthy, Sasank, Steiner, Benoit, Fang, Lu, Bai, Junjie, Chintala, Soumith, 2019. PyTorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, vol. 32, Curran Associates, Inc., pp. 8024–8035.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in python. J. Mach. Learn. Res. 12, 2825–2830.

Rusch, T. Konstantin, Bronstein, Michael M., Mishra, Siddhartha, 2023. A survey on oversmoothing in graph neural networks. ArXiv Preprint.

Salgado, Andréa M., Folly, Rossana O.M., Valdman, Belkis, Valero, Francisco, 2004. Model based soft-sensor for on-line determination of substrate. Appl. Biochem. Biotechnol. 113 (1–3), 137–144.

Schweidtmann, Artur M., Rittig, Jan G., König, Andrea, Grohe, Martin, Mitsos, Alexander, Dahmen, Manuel, 2020. Graph neural networks for prediction of fuel ignition quality. Energy Fuels 34 (9), 11395–11407.

Schweidtmann, Artur M., Zhang, Dongda, von Stosch, Moritz, 2024. A review and perspective on hybrid modeling methodologies. Digit. Chem. Eng. 10, 100136.

Shaheen, Zein, Wohlgenannt, Gerhard, Filtz, Erwin, 2020. Large Scale Legal Text Classification Using Transformer Models. arXiv:2010.12871 [cs].

Shang, Chao, Yang, Fan, Huang, Dexian, Lyu, Wenxiang, 2014. Data-driven soft sensor development based on deep learning technique. J. Process Control 24 (3), 223–233.

Sitapure, Niranjan, Kwon, Joseph Sang-Il, 2023a. CrystalGPT: Enhancing system-to-system transferability in crystallization prediction and control using time-series-transformers. Comput. Chem. Eng. 177, 108339.

Sitapure, Niranjan, Kwon, Joseph Sang-Il, 2023b. Exploring the potential of time-series transformers for process modeling and control in chemical systems: An inevitable paradigm shift? Chem. Eng. Res. Des. 194, 461–477.

Sitapure, Niranjan, Kwon, Joseph Sang-Il, 2024. Empowering hybrid models with attention-based time-series transformers: A case study in batch crystallization. In: 2024 American Control Conference. ACC, IEEE, pp. 62–67.

Sitapure, Niranjan, Sang-Il Kwon, Joseph, 2023. Introducing hybrid modeling with time-series-transformers: A comparative study of series and parallel approach in batch crystallization. Ind. Eng. Chem. Res. 62 (49), 21278–21291.

Souza, Francisco A.A., Araújo, Rui, Mendes, Jérôme, 2016. Review of soft sensor methods for regression applications. Chemometr. Intell. Lab. Syst. 152, 69–79.

Stahlberg, Felix, 2020. Neural Machine Translation: A Review and Survey. arXiv:1912.02047 [cs].

Stops, Laura, Leenhouts, Roel, Gao, Qinghe, Schweidtmann, Artur M., 2022. Flowsheet generation through hierarchical reinforcement learning and graph neural networks. AIChE J. 69 (1).

von Stosch, Moritz, Oliveira, Rui, Peres, Joana, Feyo de Azevedo, Sebastião, 2014. Hybrid semi-parametric modeling in process systems engineering: Past, present and future. Comput. Chem. Eng. 60, 86–101.

Sun, Qingqiang, Ge, Zhiqiang, 2021. A survey on deep learning for data-driven soft sensors. IEEE Trans. Ind. Inform. 17 (9), 5853–5866.

Theissen, Manfred, Wiedau, Michael, Filke, Yannik, Gomez, Luis, Hanke, Leon, Pe Ingebrigtsen, Idar, Kochmanski, David, Luukkainen, Marko, Meyer-Rössl, Reiner, Schumacher, Felix, Snijder, Paul, Temmen, Heiner, Tolksdorf, Gregor, Vazquez-Landa, David, Wagner, Axel, Welscher, Wolfgang, 2021. DEXPI P&ID specification, version 1.3. DEXPI Initiative, Version 1.3.

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, Polosukhin, Illia, 2017. Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30, Curran Associates, Inc..

Venkatasubramanian, Venkat, 2018. The promise of artificial intelligence in chemical engineering: Is it here, finally? AIChE J. 65 (2), 466–478.

Vogel, Gabriel, Hirtreiter, Edwin, Schulze Balhorn, Lukas, Schweidtmann, Artur M., 2023. SFILES 2.0: an extended text-based flowsheet representation. Optim. Eng. 24 (4), 2911–2933.

Wang, Junfu, Guo, Yuanfang, Yang, Liang, Wang, Yunhong, 2023. Enabling homogeneous GNNs to handle heterogeneous graphs via relation embedding. IEEE Trans. Big Data 9 (6), 1697–1710.

Wang, Jian-Guo, Jang, Shi-Shang, Wong, David Shan-Hill, Shieh, Shyan-Shu, Wu, Chan-Wei, 2013. Soft-sensor development with adaptive variable selection using nonnegative garrote. Control Eng. Pract. 21 (9), 1157–1164.

Wang, Hao, Liu, Ruonan, Ding, Steven X., Hu, Qinghua, Li, Zengxiang, Zhou, Hongkuan, 2024. Causal-trivial attention graph neural network for fault diagnosis of complex industrial processes. IEEE Trans. Ind. Inform. 20 (2), 1987–1996.

Wen, Qingsong, Zhou, Tian, Zhang, Chaoli, Chen, Weiqi, Ma, Ziqing, Yan, Junchi, Sun, Liang, 2022. Transformers in time series: A survey.

Xu, Keyulu, Hu, Weihua, Leskovec, Jure, Jegelka, Stefanie, 2018. How powerful are graph neural networks?. arXiv preprint arXiv:1810.00826.

Yan, Weiwu, Shao, Huihe, Wang, Xiaofan, 2004. Soft sensing modeling based on support vector machine and Bayesian model selection. Comput. Chem. Eng. 28 (8), 1489–1498.

Yi, Ling, Lu, Jun, Ding, Jinliang, Liu, Changxin, Chai, Tianyou, 2020. Soft sensor modeling for fraction yield of crude oil based on ensemble deep learning. Chemometr. Intell. Lab. Syst. 204, 104087.

Ying, Rex, Bourgeois, Dylan, You, Jiaxuan, Zitnik, Marinka, Leskovec, Jure, 2019. Gnnexplainer: Generating explanations for graph neural networks. In: Advances in Neural Information Processing Systems, vol. 32.

Yu, Yong, Si, Xiaosheng, Hu, Changhua, Zhang, Jianxun, 2019. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. Neural Comput. 31 (7), 1235–1270.

Zhao, Zhijun, Yan, Gaowei, Li, Rong, Xiao, Shuyi, Wang, Fang, Ren, Mifeng, Cheng, Lan, 2024. Instance transfer partial least squares for semi-supervised adaptive soft sensor. Chemometr. Intell. Lab. Syst. 245, 105062.

Zhu, Zhichao, Chen, Feiyang, Ni, Lei, Bian, Haitao, Jiang, Juncheng, Chen, Zhiquan, 2024. A novel transformer-based model with large kernel temporal convolution for chemical process fault detection. Comput. Chem. Eng. 188, 108762.

Zhuang, Yilin, Liu, Yixuan, Ahmed, Akhil, Zhong, Zhengang, del Rio Chanona, Ehecatl A., Hale, Colin P., Mercangöz, Mehmet, 2022. A hybrid data-driven and mechanistic model soft sensor for estimating CO2 concentrations for a carbon capture pilot plant. Comput. Ind. 143, 103747.