Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 264 (2025) 157–166

Procedia Computer Science

www.elsevier.com/locate/procedia

International Neural Network Society Workshop on Deep Learning Innovations and Applications

Adaptive Ensemble-Based Hyperparameter-Free Just-In-Time Learning for Robust Cell Culture Process Monitoring

Johnny Peng[a], Thanh Tung Khuat[a], Ellen Otte[b], Robert Bassett[b], Alistair Grevis-James[b], Katarzyna Musial[a], Bogdan Gabrys[a]

[a]*Complex Adaptive Systems Lab, University of Technology Sydney, 15 Broadway, Ultimo, NSW 2007, Australia*
[b]*CSL Innovation, 655 Elizabeth Street, Melbourne, VIC 3000, Australia*

Abstract

The production process in the biopharmaceutical industry relies on various Process Analytical Technologies (PAT) to ensure safety and consistency. Raman spectroscopy is a commonly used PAT, enabling real-time monitoring and control of cell culture processes by correlating Raman signals with key variables through data-driven models. To enhance the performance of these Raman models, just-in-time learning (JITL) can be employed to learn from a relevant subset of historical data, allowing JITL to adapt quickly to data generated from dynamic environments, such as bioreactors. This ensures the models remain accurate and effective in real-time applications. A critical and highly sensitive parameter in JITL is the number of data points used to train the model, commonly denoted as k. However, most studies determine k based on past experiences or trial and error and apply the same k value for all experiments, which is not ideal for data with constant changes, such as bioprocess data, leading to suboptimal outcomes. This study addresses this gap by developing a two-layer ensemble-based JITL method that combines predictions from models with different k values and automatically adjusts the weights of individual predictors to adapt to changes in the data, eliminating the need for manual adjustments from the user. Thirty-eight experimental bioreactor runs using different cell lines and feed media were conducted to evaluate the algorithm's performance. The findings demonstrate that the ensemble approach consistently achieved superior performance, eliminating the need to manually select and tune the highly sensitive hyperparameter k, creating a hyperparameter-free Just-In-Time Learning approach for users. This approach is particularly appealing in scenarios where hyperparameter tuning is not feasible due to limited available data and frequently changing environments requiring constant tuning of hyperparameters.

Peer-review under responsibility of the scientific committee of the IJCNN 2025

Keywords: Hyperparameter-free; Just-In-Time-Learning; Ensemble Learning; Adaptive Learning; Soft Sensor; Bioprocess; Raman Spectra

* Corresponding author. Tel.: +61-404-644-558
E-mail address: johnny.peng@student.uts.edu.au

1877-0509
Peer-review under responsibility of the scientific committee of the IJCNN 2025
10.1016/j.procs.2025.07.127

1. Introduction

Cell Culture Process Monitoring. Biopharmaceutical manufacturing is typically batch-based and involves key operations such as mammalian cell culture in bioreactors. Real-time batch process monitoring (BPM) has become a critical tool for ensuring product quality and regulatory compliance by tracking key performance indicators (KPIs) and critical quality attributes (CQAs) [11, 18, 6]. These indicators often depend on process parameters and conditions reflected in the Raman spectrum, making Raman-based soft sensors a promising approach for real-time monitoring.

Prior Work. Despite their potential, Raman-based soft sensors face challenges due to heterogeneity across runs—differences in cell lines, media, or experimental setups cause significant shifts in Raman spectral patterns. This limits model transferability and gives rise to the cold-start problem [20, 9], where pre-trained models fail to generalize due to limited comparable historical data. Just-in-time learning (JITL) has been proposed as a solution, dynamically building models from similar historical data [16, 17]. Recent improvements include time-weighted similarity [15] and distribution-based similarity measures [14].

Contribution. Although JITL effectively handles heterogeneous data, limited work has addressed the selection of its key hyperparameter, k—the number of historical data points used for training. Our study shows that model performance is highly sensitive to k, varying across target variables and base models. This presents a practical barrier, as tuning k requires significant expertise and historical data, which are often unavailable in real-world settings. Even experienced users face uncertainty when applying previous k values to new experimental configurations. To overcome these challenges, we propose a two-layer ensemble method that combines JITL models trained with varying k values. This approach eliminates the need for manual tuning and dynamically adapts ensemble weights, ensuring robust performance even in cold-start scenarios. Unlike prior ensemble JITL studies, which focus on different similarity measures [19, 21, 4], our method directly addresses the k-selection problem. It offers a practically hyperparameter-free solution, improving robustness across diverse applications with minimal data, knowledge, or tuning effort—filling a key gap in JITL research.

2. Preliminaries

2.1. Regression Problem in Cell Culture Process Monitoring

Key cell culture parameters such as glucose, lactate, and ammonia concentrations must be monitored continuously during bioreactor runs. However, these cannot be measured directly in real-time and typically rely on labour-intensive offline sampling, which is infrequent and costly. Soft sensors offer an alternative by predicting these variables from Raman spectral data, using data-driven machine learning (ML) models trained on historical data [7, 8]. To formally define this ML task, let $\mathbf{x}(t) \in \mathbb{R}^d$ denote the Raman spectrum at time t, and $y(t) \in \mathbb{R}$ be the target variable. Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i, t_i)\}_{i=1}^N$, the learning objective is to find a function f that minimises the prediction error:

$$\min_f \frac{1}{N} \sum_{i=1}^{N} \ell(f(\mathbf{x}_i, t_i), y_i) \tag{1}$$

This task faces three major challenges: (1) limited data, as offline labels are labor intensive to collect, (2) high dimensionality of Raman data (more than 3000 features), and (3) poor model transferability across runs due to variations in experimental settings, leading to the ”cold start” problem. To address this, we explore JITL, which builds a local model on the fly using only the most relevant data from past runs.

2.2. Just-In-Time Learning (JITL)

Unlike global models, JITL trains a model at prediction time using a small, relevant subset of historical data selected based on similarity to the current instance. For this study, similarity is measured using Euclidean distance:

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{n} (x_{1,i} - x_{2,i})^2}, \quad sim(\mathbf{x}_1, \mathbf{x}_2) = \exp(-d(\mathbf{x}_1, \mathbf{x}_2)) \tag{2}$$

JITL offers several advantages: it adapts to process dynamics and provides accurate predictions under non-stationary conditions, making it ideal for bioprocess monitoring [16, 15, 17, 14]. It is also competitive with other adaptive soft sensor methods across various industrial applications [10, 2, 3, 11]. When a new query Raman spectrum $\mathbf{x}_q$ at time t_q is available, JITL proceeds in two steps:

1. **Instance Retrieval:** Select the top-k most similar data points from $\mathcal{D}$:

$$\mathcal{D}_q = \{(\mathbf{x}_i, y_i, t_i) \mid i \in \text{Top-}k(sim((\mathbf{x}_q, t_q), (\mathbf{x}_i, t_i)))\} \tag{3}$$

2. **Local Model Training and Prediction:** Fit a model f_q on $\mathcal{D}_q$ and predict the output $\hat{y}_q$:

$$f_q = \arg\min_f \sum_{(\mathbf{x}_i, y_i, t_i) \in \mathcal{D}_q} \ell(f(\mathbf{x}_i, t_i), y_i), \quad \hat{y}_q = f_q(\mathbf{x}_q, t_q) \tag{4}$$

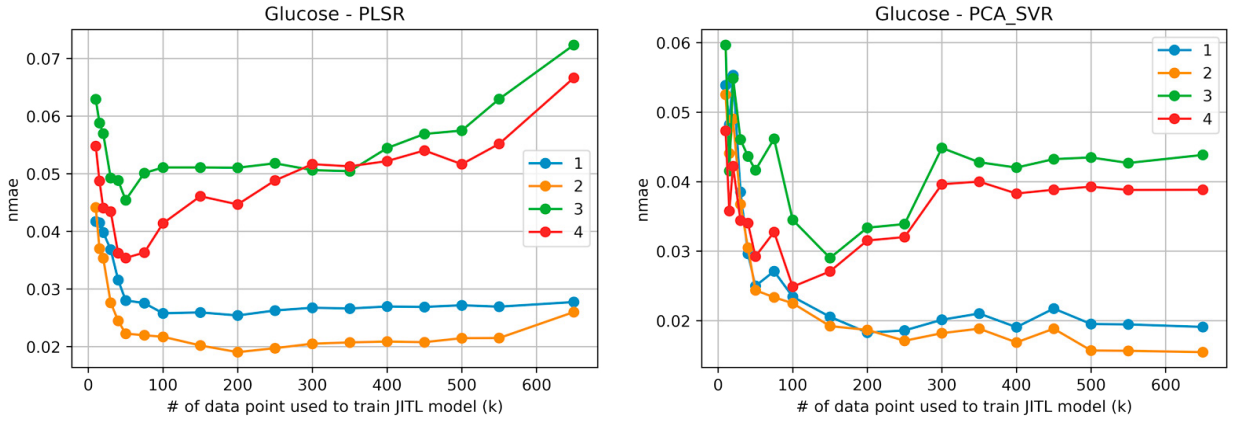
The parameter k is critical to JITL performance but often overlooked in prior work, where it is typically fixed without justification [16, 17, 15, 14]. As we will show in the next section, k is highly sensitive, and optimal values can vary significantly across runs, targets, and models.

3. Challenges of Selecting k for JITL Models

Selecting an appropriate value of k for JITL models is challenging, as the optimal value varies significantly depending on several factors. To illustrate this, we present experiments across four bioreactor runs, two base models (PLSR and SVR), and three target variables (Glucose, Lactate, Ammonia). Our findings identify three primary sources of variability that affect the choice of k: (i) differences across runs, (ii) differences in dynamics and interventions within runs, (iii) differences across target variables, and (iv) differences across the underlying model.

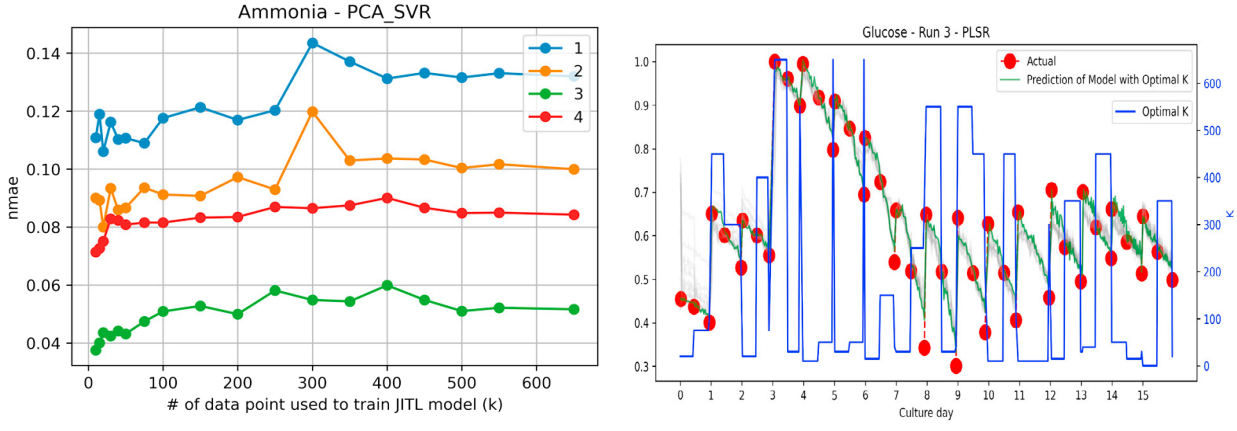
Variability Across Runs. Figure 1a shows that for Glucose prediction using PLSR, the optimal k differs across runs. Runs 3 and 4, which used a new feed media, benefit from smaller k values, while runs 1 and 2, using a familiar feed, perform better with larger k. This pattern suggests that more localised models (smaller k) are preferable when the target run is dissimilar to historical data, while broader models (larger k) perform better when the data is more homogeneous. This observation holds for both SVR and other target variables, as shown in Figures 1b and 2a, confirming that variability across runs strongly influences the choice of k.

Variability Within Runs. Figure 2b illustrates how k fluctuates significantly at different time points within a single run. No fixed k performs consistently well throughout. Notably, k tends to increase following glucose spikes caused by feed additions, and then decreases as glucose levels fall. These dynamics highlight the need for k to adapt in real time during a run, especially in response to external interventions.



(a) (b)

Fig. 1. (a) NMAE of Glucose predictions from PLSR; (b) NMAE of Glucose predictions from SVR.



(a) (b)

Fig. 2. (a) NMAE of SVR models for predicting Ammonia; (b) Optimal k at different time points for monitoring of Glucose.

Variability Across Target Variables. The optimal k also varies by target. For Glucose, good performance is achieved with k between 50 and 200, while for Ammonia, the effective range shifts to 10–50 (Figure 2a). This inconsistency complicates general k-selection across different prediction tasks.

Variability Across Models. Finally, the underlying model affects the choice of k. As seen in Figures 1a and 1b, SVR models consistently require larger k values than PLSR. This is expected due to SVR's higher complexity and capacity for modelling non-linear relationships. However, it adds yet another layer of variability to the k-selection process, making it difficult to define a one-size-fits-all value.

4. Adaptive Ensemble based Hyperparameter-Free JITL Model

As shown in earlier sections, the hyperparameter k—the number of data points retrieved to build JITL models—is highly sensitive and varies significantly across datasets, targets, and base models. This variability poses a major challenge, and while a meta-learning approach could pre-select k based on prior data, such methods are limited by the scarcity of labelled bioreactor-run data. To address this, we propose an adaptive two-layer ensemble method that dynamically adjusts prediction weights from four sub-ensembles, each using different sets of k values, as data from

the target run arrives. This approach removes the need for manual tuning and is designed to generalise across varying data distributions and models.

Each sub-ensemble integrates predictions from 18 JITL models built using different k values, ranging from 10 to all available data points (approximately 600–650). The lower k values region is more densely sampled as the sensitivity is higher when k is low. (as shown in Fig 1a and Fig 1b). These values cover a broad range of proportions (1.5% to 100%) of the dataset and provide a practically hyperparameter-free setup for users. While users can extend or refine this list, the ensemble automatically adjusts model weights, eliminating the need for manual tuning. **The first sub-ensemble** is built before the target run using cross-validation on historical data, resulting in $\mathcal{K}_{cv}$ and $\mathcal{M}_{cv}$, which are the set of k values and corresponding JITL models that constitute the ensemble, respectively. It performs well when the target run resembles past runs but struggles when distributions differ. To adapt, a **second sub-ensemble** $\mathcal{M}_{current}$ is constructed during the run using all accumulated data, updating $\mathcal{K}_{current}$ as new labels arrive. While adaptive, it may lag in responding to sudden distribution shifts. To handle such shifts, we introduce a **third sub-ensemble** $\mathcal{M}_{last}$, optimised using only the most recent observations. This enhances responsiveness to local data changes within a run. Since the previous three sub-ensembles rely on empirical performance and are prone to overfitting, we add a **fourth sub-ensemble** $\mathcal{M}_{sim}$, which selects k based on the similarity between the most recent observation and the training data. This method offers additional regularisation by avoiding direct optimisation on target data. Lastly, the predictions from these four ensemble models are combined to form the **second-layer ensemble prediction** via a simple adaptive ranking system. Initially, all models start with zero weight. After each new label, the models are ranked by prediction error, and weights are incremented by 3, 2, 1, or 0 accordingly. This ensemble-of-ensembles framework provides a robust and adaptive solution that generalises across varying targets, models, and data distributions.

To illustrate the learning process of each sub-ensemble, we will formally denote $\mathcal{K} = \{k_1, k_2, \ldots, k_m\}$ the set of selected k values. For each $k_i \in \mathcal{K}$, a model M_{k_i} is trained using the k_i-nearest points to the query instance, forming the model set $\{M_{k_1}, M_{k_2}, \ldots, M_{k_m}\}$. As the query instance changes, this set is updated to reflect the new neighbourhood, allowing the ensemble to capture patterns at multiple local granularities. The detailed steps for calculating the weights for each sub-ensemble are discussed in the subsections below.

4.1. To find $\mathcal{M}_{cv}$

Prior to the target run start, 5-fold cross-validation will be used on the historical data to determine $\mathcal{K}_{cv}$ and corresponding binary ensemble weights (0 or 1) of models $\{M_{k_1}, M_{k_2}, \ldots, M_{k_m}\}$. The models with the weight of 1 will form a part of the model combination for $\mathcal{M}_{cv}$. To compute the weights, we first divde the historical dataset $\mathcal{H} = \{(\mathbf{x}_i, y_i)\}$ into 5 equally sized folds $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_5$, where $\mathbf{x}_i$ represents the historical features, and y_i represents the corresponding offline measurement. Then, for each subset $\mathcal{M} \subseteq \{M_{k_1}, M_{k_2}, \ldots, M_{k_m}\}$, use four folds for training and construct the ensemble prediction $\hat{P}_{\mathcal{M}}$ and corresponding test error $E_{\mathcal{M}}$ on the remaining fold. Repeat this process for all five-fold combinations and compute the average cross-validation error $\bar{E}_{\mathcal{M}}$, and the subset that minimises the average cross-validation error will be the $\mathcal{M}_{cv}$. Correspondingly, the average of predictions from $\mathcal{M}_{cv}$ will be denoted as $P_{\mathcal{M}_{cv}}$.

$$\hat{P}_{\mathcal{M}} = \frac{1}{|\mathcal{M}|}\sum_{M_k \in \mathcal{M}} P_k, \quad E_{\mathcal{M}} = \frac{1}{|\mathcal{H}_{\text{test}}|}\sum_{(\mathbf{x}_j, y_j) \in \mathcal{H}_{\text{test}}} \left|\hat{P}^{(j)}_{\mathcal{M}} - y_j\right|, \quad \bar{E}_{\mathcal{M}} = \frac{1}{5}\sum_{i=1}^{5} E^{(i)}_{\mathcal{M}}, \quad \mathcal{M}_{cv} = \arg\min_{\mathcal{M}} \bar{E}_{\mathcal{M}} \tag{5}$$

4.2. To find $\mathcal{M}_{current}$ and $\mathcal{M}_{last}$

Perform an exhaustive search to evaluate all possible combinations of models from $\{M_{k_1}, M_{k_2}, \ldots, M_{k_m}\}$. For each subset $\mathcal{M} \subseteq \{M_{k_1}, M_{k_2}, \ldots, M_{k_m}\}$, calculate the corresponding ensemble prediction $\hat{P}_{\mathcal{M}}$ as defined in Equation 5. Then, let $\mathcal{F}$ be a set of feedback accumulated from the current run that contains n elements, with each $f_i = \{(\mathbf{x}_i, y_i)\}$. Find subset $\mathcal{M}_{current}$ that minimises the total prediction error with respect to all feedback from the current run $\mathcal{F}$, and the subset $\mathcal{M}_{last}$ that minimises the total prediction error with respect to the last feedback from the current run $\mathcal{F}$. Correspondingly, the average of predictions from $\mathcal{M}_{current}$ and $\mathcal{M}_{last}$ will be denoted as $P_{current}$ and P_{last}, respectively.

This step can be formally defined as equation 6 below:

$$\mathcal{F} = \{f_1, f_2, \ldots, f_n\}, \quad \mathcal{M}_{current} = \arg\min_{\mathcal{M}} \sum_{j=1}^{n} \left|\hat{P}_{\mathcal{M}}^{(j)} - f_j\right|, \quad \mathcal{M}_{last} = \arg\min_{\mathcal{M}} \left|\hat{P}_{\mathcal{M}}^{(j)} - f_n\right| \tag{6}$$

4.3. To find $\mathcal{M}_{sim}$

1. **Compute Data-to-Dataset Similarity for Each Data Point in Historical Data**: For each data point $\mathbf{x}_i$ in the historical training dataset $\mathcal{D}$, calculate its pairwise similarity $S(\{\mathbf{x}_i, \mathbf{y}_i\}, \{(\mathbf{x}_j, \mathbf{y}_j\})$ with every other data point $\mathbf{x}_j \in \mathcal{D}, j \neq i$. The similarity is computed as:

$$S(\{\mathbf{x}_i, \mathbf{y}_i\}, \{(\mathbf{x}_j, \mathbf{y}_j\}) = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|^2\right) \cdot \text{scaling\_factor}, \quad \text{scaling\_factor} = \begin{cases} 1 - \frac{|y_i - y_j|}{y_{\max} - y_{\min}}, & \text{if } \mathcal{F} \neq \{\} \\ 1, & \text{if } \mathcal{F} = \{\} \end{cases} \tag{7}$$

   Here, $y_{\max}$ and $y_{\min}$ are the maximum and minimum labels in the dataset, respectively. This ensures that the similarity computation is adjusted by both feature and label consistency when labels are available. If feedback from the current run is unavailable ($\mathcal{F} = \{\}$), then the scaling factor defaults to 1, and the similarity relies entirely on feature-based computation.
2. **Compute Average Similarity to Historical Data for Last Feedback**: For the last feedback f_n, calculate its pairwise similarity $S(\{\mathbf{x}_{f_n}, f_n\}, \{\mathbf{x}_i, \mathbf{y}_i\})$ with each data point $\mathbf{x}_i \in \mathcal{D}$ using Eq. 7. If $\mathcal{F} = \{\}$, calculate $S(\mathbf{x_q}, \mathbf{x}_j)$ instead, where $\mathbf{x_q}$ is the inferencing query data. The similarity of $\{\mathbf{x}_{f_n}, f_n\}$ or $\mathbf{x}_q$ to the dataset $\mathcal{D}$, denoted as $S_{\text{avg}}(\{\mathbf{x}_{f_n}, f_n\}, \mathcal{D})$ and calculated as the average of pairwise similarities to data points in $\mathcal{D}$:

$$S_{\text{avg}}(\{\mathbf{x}_{f_n}, f_n\}, \mathcal{D}) = \begin{cases} \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}_i \in \mathcal{D}} S(\{\mathbf{x}_{f_n}, f_n\}, \{\mathbf{x}_i, \mathbf{y}_i\})), & \text{if } \mathcal{F} \neq \{\} \\ \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}_i \in \mathcal{D}} S(\mathbf{x_q}, \mathbf{x}_j), & \text{if } \mathcal{F} = \{\} \end{cases} \tag{8}$$

3. **Determine k_{sim} Based on Similarity Percentile**: Using the distribution of data-to-dataset similarities $S(\{\mathbf{x}_i, \mathbf{y}_i\}, \{(\mathbf{x}_j, \mathbf{y}_j\}) \mid \mathbf{x}_i \in \mathcal{D}, i \neq j)$, determine the percentile p of $S_{\text{avg}}(\{\mathbf{x}_{f_n}, f_n\}, \mathcal{D})$ within this distribution and define k_{sim}, $\mathcal{M}_{sim}$ as equation 9 . Correspondly, the average prediction from $\mathcal{M}_{sim}$ will be denoted as P_{sim}.

$$k_{sim} = \arg\min_{k \in \mathcal{K}} |k - \text{int}(p \cdot |\mathcal{D}|)|, \quad \mathcal{M}_{sim} = \{M_{k_{sim}}\} \tag{9}$$

4.4. Combining $P_{cv}, P_{current}, P_{last}, P_{sim}$

As the weight calculation for $P_{current}, P_{last}$ depends on the availability of feedback, different approaches will be used to combine the predictions from individual ensemble models to form the final prediction $P_{ensemble}$ at different stages of the run as shown below. In addition, to provide more clarity on the high-level workflow of the proposed method, algorithm 1 consolidates all discussed steps into a pseudo-code.

1. **Prior to receiving the first feedback**: Given that we do not have feedback at this stage, we cannot find $\mathcal{M}_{current}, \mathcal{M}_{last}$. Thus, the only predictions we can make at this stage are P_{cv} and P_{sim}, which will be averaged to form the final prediction $P_{ensemble}$.
2. **After receiving the first feedback and before receiving the second feedback**: At this stage, we can form all four predictors $P_{cv}, P_{current}, P_{last}, P_{sim}$, however to determine the weight for the two-layer ensemble, we will need the second feedback. Thus, we first compare the accuracy of P_{cv}, P_{sim} based on the first feedback. Then, the

average of P_{last} and the winner between P_{cv} and P_{sim} will form the final prediction $P_{ensemble}$. Note that we did not include $P_{current}$ in the averaging as it is the same as P_{last} at this stage.

3. **After receving the second feedback**: The weights for individual ensemble models will be initialised to zero. After the second feedback is received, weights will be added to individual ensemble models based on the absolute error of the models. Weights of 3, 2, 1, 0 will be added to the weights of models that ranked 1st, 2nd, 3rd, and 4th, respectively. These weights are accumulative as weight is added every time new feedback becomes available, and the final prediction $P_{ensemble}$ will be the weighted average of $P_{cv}, P_{current}, P_{last}, P_{sim}$ based on the weights (normalized by total weight) of the corresponding ensemble model at the time of the prediction.

Algorithm 1 Adaptive Ensemble-Based Hyperparameter-Free JITL Model

Require: Predefined set of k values $\mathcal{K}$, Historical data $\mathcal{H}$, Target run data $\mathcal{T}$
Ensure: Final prediction $P_{ensemble}$

Set ensemble weights: $\{w_{cv}, w_{current}, w_{last}, w_{sim}\} \leftarrow 0$
Determine hyperparameters for JITL models:
for $k \in \mathcal{K}$ **do**
    Use *Optuna* to optimize hyperparameters for M_k based on 5-fold cross-validation on $\mathcal{H}$.
    Save the optimal hyperparameters for M_k.
end for
Determine $\mathcal{K}_{cv}$ using 5-fold cross-validation on $\mathcal{H}$.
$\mathcal{F} \leftarrow \varnothing$
while *new data* $\mathbf{q}$ arrives in $\mathcal{T}$ **do**
    for $k \in \mathcal{K}$ **do**
        Retrieve the k-nearest data points to $\mathbf{q}$ from $\mathcal{H} \cup \mathcal{F}$
        Construct M_k using the retrieved data points and the pre-optimised hyperparameters
        Compute prediction P_k for $\mathbf{q}$ using M_k, then discard M_k to release memory.
    end for
    Construct ensemble predictions:

    1. $P_{cv} \leftarrow$ Average of $\{P_k : k \in \mathcal{K}_{cv}\}$
    2. Determine $\mathcal{K}_{sim}$ based on the similarity of f_n or $\mathbf{q}$ to $\mathcal{H} \cup \mathcal{F}$; $P_{sim} \leftarrow P_k : k \in \mathcal{K}_{sim}$
    3. If $|\mathcal{F}| > 0$ Determine $\mathcal{K}_{current}$ by optimizing over $\mathcal{F}$; $P_{current} \leftarrow$ Average of $\{P_k : k \in \mathcal{K}_{current}\}$
    4. If $|\mathcal{F}| > 0$ Determine $\mathcal{K}_{last}$ using the most recent data in $\mathcal{F}$; $P_{last} \leftarrow$ Average of $\{P_k : k \in \mathcal{K}_{last}\}$

    if $|\mathcal{F}| = 0$ **then**
        $P_{ensemble} \leftarrow$ Average of P_{cv} and P_{sim}
    else if $|\mathcal{F}| = 1$ **then**
        $P_{ensemble} \leftarrow$ Average of best(P_{cv}, P_{sim}) and P_{last}
    else
        $P_{ensemble} \leftarrow$ Weighted average of ensemble predictions $\{P_{cv}, P_{current}, P_{last}, P_{sim}\}$
        if *Label* for $\mathbf{q}$ is available **then**
            Rank $\{P_{cv}, P_{current}, P_{last}, P_{sim}\}$ based on its absolute difference to label, then update weights $\{w_{cv}, w_{current}, w_{last}, w_{sim}\}$ accordingly.
        end if
    end if
    if *Label* for $\mathbf{q}$ is available **then**
        $\mathcal{F} \leftarrow \mathcal{F} \cup \{\mathbf{q}\}$
    end if
end while

5. Data & Experimental Setting

Datasets. This study uses data from 38 5L-bioreactor runs, each with varying cell lines and feeds. Each run spans 2 weeks, producing 10–40 offline measurements for Glucose, Lactate, and Ammonia. The goal is to predict these metabolites from Raman spectra (3325 dimensions, 100–3425 cm^{-1}). As Raman and offline data are collected at different times, the offline measurements are appropriately mapped to Raman spectra. Models are evaluated using leave-one-batch-out cross-validation, with each batch including 2–4 parallel runs with similar configurations. During training, JITL models can access training data and feedback collected during the validation/test run at inference time.

Preprocessing. To reduce noise, Raman spectra are trimmed to 500–3000 cm^{-1}, smoothed using a Savitzky–Golay filter (window=25, order=1, derivative=1), and normalised using standard normal variate (SNV). This preprocessing enhances signal quality, as guided by Poth et al. [13].

Prediction Models. We evaluate the approach using PLSR (linear) and SVR (non-linear), both of which are widely used in bioprocess monitoring. This allows us to assess robustness across model complexities and understand how model type influences the optimal k range.

Dimensionality Reduction. For SVR, dimensionality is reduced using principal component analysis (PCA) before model fitting. PLSR performs inherent dimensionality reduction and uses the preprocessed Raman data directly.

Hyperparameter Tuning. Model and dimensionality reduction hyperparameters are optimised via the Tree-structured Parzen Estimator from the *Optuna* toolbox [1], using 100 trials on 5-fold CV splits of the training data. The best configuration minimises average RMSE across folds.

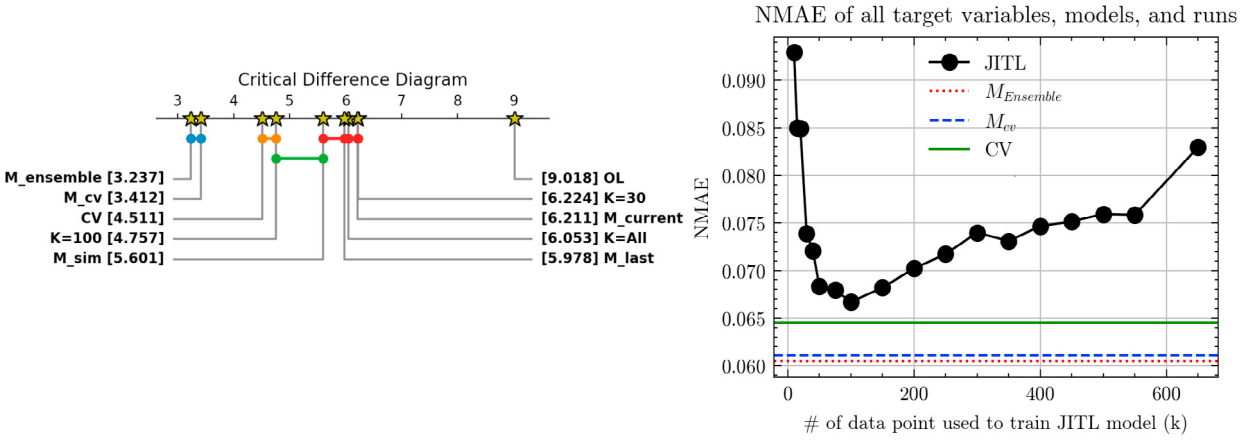
6. Results and Discussion

To benchmark the proposed two-layer ensemble method, we compare it against individual ensemble variants and baseline methods. These baselines include selecting k via 5-fold cross-validation, as well as setting k to fixed values of $k = 30$ and $k = 100$ as the prior studies [16, 15, 17, 14] did. We follow the model comparison method recommended by [5], using Friedman and Nemenyi post-hoc tests and visualised results with Critical Difference Plots (CDP).

Overall Performance of Methods. Figure 3a presents the average rank of each method across 228 experiments, covering three target variables, two models (PLSR, SVR), and 38 bioreactor runs. In addition to JITL variants, we include online learning models, recursive PLSR, and online SVR [12], as well as batch learning (k = All), for a broader comparison. Results group the methods into four performance tiers. The two-layer ensemble and M_{cv} form the top-performing group. While the ensemble ranks higher, the difference is not statistically significant. The second group includes the cross-validation-based single k, and $k = 100$, both outperforming individual ensemble variants (except M_{cv}), OL, batch learning, and $k = 30$. Notably, $k = 100$ performs best among fixed k values (Figure 3b), but still lags behind adaptive methods, underscoring the limitations of static k choices. Online learning performs worst, likely due to its order dependency and poor generalisation when historical and current data differ. Its use of all prior data without clear transitions between runs may have degraded accuracy.

Performance by Target and Model. As shown in Table 1a, the two-layer ensemble achieves the lowest NMAE in 4 of 6 task-model combinations and ranks second in the others, demonstrating robust performance. M_{cv} also performs consistently well, especially for Glucose, suggesting greater inter-run similarity of Glucose.

Robustness to Cold-Start Scenario. Runs 3 and 4 represent true cold-start scenarios, utilising a novel feed composition not present in other runs. Table 1b shows that the two-layer ensemble and $M_{current}$ perform best. $M_{current}$ adapts based on feedback within the current run, making it resilient to distribution shifts. M_{last}, which relies on the most recent observation, also performs well but shows instability due to its limited context. In contrast, M_{cv} performs poorly here, as it is ensembled based on historical data and assumes inter-run similarity. These results reinforce the value of dynamic feedback-driven approaches in unseen scenarios, with the two-layer ensemble adapting effectively through weighted model updates.



(a) Critical Difference Plot across all experiments

(b) NMAE of different fixed values of k

Fig. 3. Aggregated Performance of Methods

(a) NMAE by target and model

NMAE	PCA_SVR			PLSR		
Method	Ammonia	Glucose	Lactate	Ammonia	Glucose	Lactate
CV	9.716%	4.084%	3.950%	10.549%	5.008%	5.377%
K=100	9.820%	4.185%	4.265%	11.671%	4.731%	5.341%
K=30	*9.558%*	6.239%	5.540%	10.503%	6.608%	5.869%
M_current	10.077%	10.573%	4.284%	10.674%	11.500%	5.041%
M_cv	9.586%	**3.921%**	*3.874%*	*10.225%*	**4.309%**	*4.745%*
M_ensemble	**9.345%**	*3.960%*	**3.866%**	**10.060%**	*4.502%*	**4.537%**
M_last	9.885%	11.018%	4.243%	10.434%	11.248%	4.799%
M_sim	9.861%	4.371%	4.212%	15.748%	5.341%	5.800%

(b) NMAE for cold-start runs (Runs 3 & 4)

NMAE	PCA_SVR			PLSR			Average	Average
Method	Ammonia	Glucose	Lactate	Ammonia	Glucose	Lactate	All	Rank
CV	6.197%	4.139%	3.116%	5.881%	5.114%	3.822%	4.712%	5.8
K=100	6.620%	**2.969%**	2.864%	7.848%	4.624%	3.277%	4.700%	4.5
K=30	6.263%	4.022%	3.518%	6.237%	4.635%	4.416%	4.848%	6.3
M_current	**6.105%**	3.447%	*2.552%*	5.725%	4.514%	*3.072%*	*4.236%*	**2.3**
M_cv	6.453%	3.481%	2.915%	**5.305%**	4.662%	3.327%	4.357%	4.7
M_ensemble	6.212%	3.213%	2.650%	5.775%	*4.358%*	3.073%	**4.213%**	*3.0*
M_last	*6.152%*	4.011%	**2.392%**	6.452%	5.170%	**2.685%**	4.477%	4.0
M_sim	7.103%	*3.114%*	3.493%	9.455%	**4.315%**	3.553%	5.172%	5.3

Table 1. Performance breakdown by (a) prediction task and model, and (b) cold-start runs

7. Conclusion and Future Research Directions

This study investigated the impact of the hyperparameter k on the performance of JITL models for cell culture process monitoring—an area previously underexplored. Results reveal that the optimal k varies across runs depending on the similarity between historical and target batches. Less localised models perform better with high similarity and vice versa. Moreover, the optimal k shifts over the course of bioreactor runs and is sensitive to external interventions, highlighting k's critical role in model performance. To address this, we proposed a two-layer ensemble method that eliminates the need to manually select k by combining predictions from multiple JITL models with different k values. This method dynamically adjusts ensemble weights based on current data, offering a robust, adaptive, and practically hyperparameter-free solution for practitioners.

The ensemble approach demonstrated strong performance across various target variables, models, and run conditions, including cold-start scenarios. Its primary drawback is computational complexity, as training multiple models increases runtime—though this can be mitigated through parallelization. In settings where feedback is unavailable or computational resources are constrained, a viable alternative is the M_{cv} approach, which selects an ensemble of k values based on cross-validation. While faster, it is less adaptive in cold-start situations since it relies solely on historical data. Among non-ensemble methods, selecting k via cross-validation outperformed fixed values, offering better adaptability without significantly increasing complexity.

For future research, broader validation of the proposed ensemble method is needed across diverse base models, offline measurements, and application domains, including scenarios with limited feedback availability to extend its practical applicability. Lastly, binary ensemble weights are used in this study for simplicity. Thus, future studies could experiment with continuous ensemble weights, which may further improve the performance of the proposed method.

Acknowledgements

This research was supported under the Australian Research Council's Industrial Transformation Research Program (ITRP) funding scheme (project number IH210100051). The ARC Digital Bioprocess Development Hub is a collaboration between The University of Melbourne, University of Technology Sydney, RMIT University, CSL Innovation Pty Ltd, Cytiva (Global Life Science Solutions Australia Pty Ltd) and Patheon Biologics Australia Pty Ltd.

References

[1] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M., 2019. Optuna: A next-generation hyperparameter optimization framework. CoRR abs/1907.10902. URL: http://arxiv.org/abs/1907.10902, arXiv:1907.10902.

[2] Bakirov, R., Gabrys, B., Fay, D., 2015. On sequences of different adaptive mechanisms in non-stationary regression problems, in: 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. doi:10.1109/IJCNN.2015.7280779.

[3] Bakirov, R., Gabrys, B., Fay, D., 2017. Multiple adaptive mechanisms for data-driven soft sensors. Computers & Chemical Engineering 96, 42–54. doi:10.1016/j.compchemeng.2016.08.017.

[4] Chen, X., Zhao, J., Xu, M., Yang, M., Wu, X., 2023. A quality prediction method based on tri-training weighted ensemble just-in-time learning–relevance vector machine model. Processes 11, 3129–.

[5] Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30. URL: http://jmlr.org/papers/v7/demsar06a.html.

[6] Jiang, M., Severson, K.A., Love, J.C., Madden, H., Swann, P., Zang, L., Braatz, R.D., 2017. Opportunities and challenges of real-time release testing in biopharmaceutical manufacturing. Biotechnol Bioeng 114, 2445–2456.

[7] Kadlec, P., Gabrys, B., 2009a. Architecture for development of adaptive on-line prediction models. Memetic Computing 1, 241–269. doi:10.1007/s12293-009-0017-8.

[8] Kadlec, P., Gabrys, B., 2009b. Soft sensors: where are we and what are the current and future challenges? IFAC Proceedings Volumes 42, 572–577. doi:10.3182/20090921-3-tr-3005.00098.

[9] Kadlec, P., Gabrys, B., 2010. Adaptive on-line prediction soft sensing without historical data, in: The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. doi:10.1109/IJCNN.2010.5596965.

[10] Kadlec, P., Grbić, R., Gabrys, B., 2011. Review of adaptation mechanisms for data-driven soft sensors. Computers & Chemical Engineering 35, 1–24. doi:10.1016/j.compchemeng.2010.07.034.

[11] Khuat, T.T., Bassett, R., Otte, E., Grevis-James, A., Gabrys, B., 2024. Applications of machine learning in antibody discovery, process development, manufacturing and formulation: Current trends, challenges, and opportunities. Computers & Chemical Engineering 182, 108585. doi:10.1016/j.compchemeng.2024.108585.

[12] Ma, J., Theiler, J., Perkins, S., 2003. Accurate on-line support vector regression. Neural Computation 15, 2683–2703. doi:10.1162/089976603322385117.

[13] Poth, M., Magill, G., Filgertshofer, A., Popp, O., Großkopf, T., 2022. Extensive evaluation of machine learning models and data preprocessings for raman modeling in bioprocessing. Journal of Raman Spectroscopy 53, 1580–1591. doi:https://doi.org/10.1002/jrs.6402, arXiv:https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/pdf/10.1002/jrs.6402.

[14] Rashedi, M., Khodabandehlou, H., Wang, T., Demers, M., Tulsyan, A., Garvin, C., Undey, C., 2024. Integration of just-in-time learning with variational autoencoder for cell culture process monitoring based on raman spectroscopy. Biotechnol Bioeng 121, 2205–2224.

[15] Tulsyan, A., Khodabandehlou, H., Wang, T., Schorner, G., Coufal, M., Undey, C., 2021. Spectroscopic models for real-time monitoring of cell culture processes using spatiotemporal just-in-time gaussian processes. AIChE Journal 67. doi:10.1002/aic.17210.

[16] Tulsyan, A., Schorner, G., Khodabandehlou, H., Wang, T., Coufal, M., Undey, C., 2019. A machine-learning approach to calibrate generic raman models for real-time monitoring of cell culture processes. Biotechnology and Bioengineering 116, 2575–2586. doi:https://doi.org/10.1002/bit.27100, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/bit.27100.

[17] Tulsyan, A., Wang, T., Schorner, G., Khodabandehlou, H., Coufal, M., Undey, C., 2020. Automatic real-time calibration, assessment, and maintenance of generic raman models for online monitoring of cell culture processes. Biotechnology and Bioengineering 117, 406 – 416. doi:10.1002/bit.27205.

[18] Undey, C., Ertunç, S., Mistretta, T., Pathak, M., 2009. Applied advanced process analytics in biopharmaceutical manufacturing: Challenges and prospects in real-time monitoring and control. IFAC Proceedings Volumes 42, 177–182. URL: https://www.sciencedirect.com/science/article/pii/S147466701530269X, doi:https://doi.org/10.3182/20090712-4-TR-2008.00026. 7th IFAC Symposium on Advanced Control of Chemical Processes.

[19] Wei, L., Zhai, B., Sun, H., Hu, Z., Zhao, Z., 2022. An ensemble jitl method based on multi-weighted similarity measures for cold rolling force prediction. ISA transactions 126, 326–337.

[20] Weiss, K., Khoshgoftaar, T.M., Wang, D., 2016. A survey of transfer learning. Journal of Big Data 3, 9. URL: https://doi.org/10.1186/s40537-016-0043-6, doi:10.1186/s40537-016-0043-6.

[21] Yuan, X., Zhou, J., Wang, Y., Yang, C., 2018. Multi-similarity measurement driven ensemble just-in-time learning for soft sensing of industrial processes. Journal of chemometrics 32.