Comparison of decision support systems for optimisation of industrial penicillin production

Gregor Mlinarič
University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia
gregor.mlinaric@outlook.com

Miha Glavan
Jožef Stefan Institute, Ljubljana, Slovenia
miha.glavan@ijs.si

Lovro Šubelj
University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia
lovro.subelj@fri.uni-lj.si

***Abstract*—Industry 5.0 emphasizes enhanced human–machine collaboration, where digital solutions support operators in complex processes. This work investigates three families of data-driven decision support systems to assist operators in managing the penicillin production process. The first relies on predictive modelling and optimisation, the second on graph neural networks, and the third combines both into a hybrid approach. Experimental results show that all proposed systems outperform predefined recipes in most cases, with hybrid models achieving the best overall performance. The findings highlight the importance of selecting an appropriate decision support strategy aligned with production objectives.**

***Index Terms*—Decision support systems, Hybrid modelling, Recommender systems, Bioprocess optimization, Penicillin production, Process monitoring and control, Industry 5.0, Graph neural networks, XGBoost, IndPenSim.**

I. Introduction

With the introduction of Industry 5.0 guidelines, the development of technological solutions is increasingly oriented towards enhanced human-system integration. Recent trends promote social sustainability by ensuring that human roles are supported rather than replaced [1]. One approach toward this goal represents the adoption of systems that help production operators to manage their tasks by monitoring their activities and suggesting appropriate corrective actions. Such systems improve the working conditions either by providing learning support to less experienced operators, or even guide and safeguard more experienced ones.

One concrete example of such systems are *decision support systems*, interactive information systems that integrate data, analytical modelling techniques, and user interfaces to support human decision-makers in addressing complex tasks [2]. They are used across a wide range of domains, including production scheduling [5], transportation route planning [4], and clinical decision support [6]. However, their application in bioprocess operations remains limited, despite the complexity and strong dependence of such processes on operator interventions.

Decision support systems can be grouped into several categories, such as *model-driven* and *knowledge-driven* DSS. However, for our purposes the most relevant are *data-driven* DSSs [3]. These systems recommend operator actions based on extensive historical data, statistical analysis, and machine-learning techniques. Their main advantage is that they are largely model-agnostic and require little explicit domain modelling. Nevertheless, their performance strongly depends on the quantity and quality of the available data, and insufficient or noisy measurements can limit their effectiveness. This motivates the development of hybrid approaches that combine the strengths of different data-driven models.

In this paper three approaches were developed and evaluated using the *IndPenSim* [9] simulation environment for industrial penicillin production. The first approach is based on *optimization using predictive models*. The models are first trained on historical data of *IndPenSim* and then used together with appropriate optimization method to determine the optimal input variables. The second approach is based on the development of recommender system using *graph neural networks* (GNNs). This approach leverages graph-based representations of the training data and neural architectures designed to operate on such structures [8]. The last hybrid approach, generates decision recommendations by combining the knowledge of both previous approaches. Together, these approaches aim to provide actionable, data-driven support to operators and address gaps in current bioprocess decision-support methodologies.

II. Case Study

Goldrick et al. introduced the *IndPenSim* simulation framework. The simulation model was designed to provide a realistic representation of an industrial bioreactor used in penicillin production [9]. Based on this environment, a training dataset consisting of 6400 simulated cases was constructed. The resulting process attributes can be organized into the following categories.

- *State variables* (35 variables) capture the simulated measurements of the production process at time t. Redundant attributes and those corresponding to internal, non-observable states of the simulator were excluded. Examples of retained state variables include the cooling flow rate F_c, dissolved oxygen concentration DO_2, and carbon evolution rate CER. They are collected in the vector
$$\mathbf{s}_t = [s_t^{(1)}, s_t^{(2)}, \ldots, s_t^{(34)}, s_t^{(35)}]. \quad (1)$$
- *Manipulative variables* (11 variables) correspond to quantities that the operator can directly adjust to increase penicillin yield. For the purposes of this study, the attention is restricted to to six key variables to that have

shown the strongest control effects: *temperature* T, *sugar feed rate* F_s, *phenylacetic acid flow* F_{PAA}, *oil flow* F_{oil}, *air flow* F_g, and *water flow* F_w. At time t, these are represented by the vector

$$\mathbf{a}_t = [T^{(t)}, F_s^{(t)}, F_{PAA}^{(t)}, F_{oil}^{(t)}, F_g^{(t)}, F_w^{(t)}]. \quad (2)$$

- The *target variables*, $y_{increase,t}$ and $y_{total,t}$, characterize the change in produced penicillin mass throughout the experiment. The variable $y_{increase,t}$ denotes the mass increment since the previous time step, whereas $y_{total,t}$ specifies the cumulative penicillin mass, produced up to time t. The final yield at the end of batch is denoted by $y_{total,final}$.

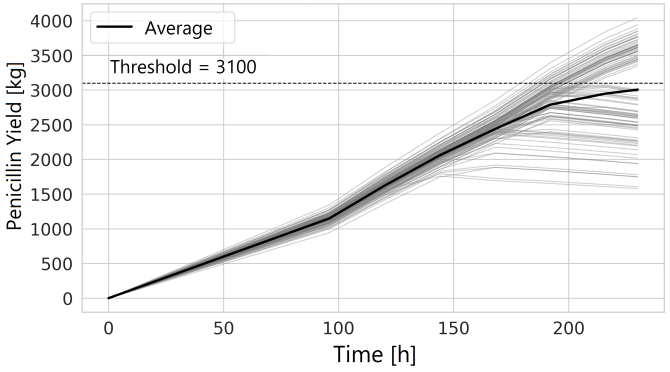

Fig. 1. Growth of pencilline over time using a predefined recipe in 100 experiments.

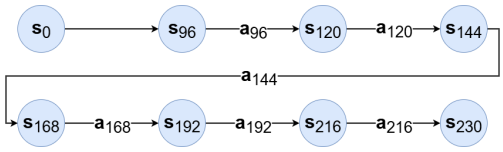

Fig. 2. The diagram shows when the operator influences the manipulative variables described by $\mathbf{a}_t$. At time $t \in \{96, 120, 144, 168, 192, 216\}$ the operator performs action $\mathbf{a}_t$ based on the current state described by $\mathbf{s}_t$. The duration of production is limited to 230 hours.

In industrial settings, process control is commonly carried out using a predefined *recipe* which is based on the past practical experience and specifies the adaptation of manipulative variables during the evolution of one batch. During batch operation, the operators are responsible for continuously monitoring the process state and readjust the recipe values whenever needed in order to improve the operation conditions and final yield of the process. As shown in Figure 1, the same recipe can lead to markedly different outcomes. This highlights the importance of operators, that need to constantly monitor and dynamically readjust the manipulative variables according to their best knowledge.

The objective of this study was to evaluate the effectiveness of several methods for monitoring and readjusting the manipulative variables within the *IndPenSim* environment. In our experimental setup, it was assumed that the operator could adjust the variables represented by $\mathbf{a}_t$ at predefined time points (as illustrated in Figure 2) in order to maximize the final penicillin yield $y_{total,final}$. Moreover, it was assumed that the operator accepted the recommended actions and applied them to the process.

A. Generating the historical process operation dataset

In practice the decision support system would be developed on the basis of the historical production data. With this, the current knowledge and practice of the operators would be considered when training the decision support logic. But for the addressed process there was no historical process data or described operators' logic that could be used as a basis. Consequently, the dataset of historical process operation was generated using the established IndPenSim API interface. A summary of the data is presented in Table I.

The data were produced using three distinct strategies:

- *Predefined recipe (R):* 3200 experiments in which the operator followed a fixed recipe and performed no custom interventions.
- *Monte Carlo walk (MC):* 1600 experiments where, at each decision point, the operator randomly selected one action from the set of allowed actions.
- *Greedy walk (GW):* 1600 experiments where the operator chose the best action from the set of allowed actions at each decision point.

The set of possible actions was defined as follows: at predefined time points, the operator was allowed to modify only a small subset of the manipulative variables. In most cases, a single variable was adjusted, while in a smaller number of cases two variables were altered simultaneously. All remaining manipulative variables followed the values specified by the recipe.

The subset of variables selected for intervention remained fixed throughout each individual experiment, whereas the set of permissible actions was updated based on the most recently executed action. At each decision point at time t, the operator could choose among five candidate values for the selected variable $a_i^{(t)} \in \mathbf{a}_t$:

$$\{m - 2k,\ m - k,\ m,\ m + k,\ m + 2k\} \quad (3)$$

where m denotes the value of the manipulative variable a_i at the previous time step, and k represents the step size (a variable-specific discrete increment). In cases where two variables were manipulated simultaneously, the available choices for each variable were independent of one another.

The experiments were additionally divided into two subgroups according to the time at which the operator's interventions began (at $t = 0$ h or $t = 96$ h).

For comparison, all experiments were also repeated using the same random-number-generator seeds, but this time the process followed the standard production recipe without any operator intervention.

TABLE I
STATISTICAL SUMMARY OF THE FINAL YIELDS $y_{total,final}$ BY METHOD FOR THE TRAINING DATASET.

Method	No. of exp.	Mean $[kg]$	Median $[kg]$	Std. dev. $[kg]$	Min $[kg]$	Max $[kg]$
MC	1600	2403	2541	1252	0	4715
GW	1600	3309	3548	652	1474	4900
R	3200	3050	3390	654	1127	4207
Total	6400	2953	3386	908	0	4900

III. METHODOLOGY

During the batch operation, the values of the manipulative variables in $\mathbf{a}_t$ must be selected at predefined time points based on the currently available simulated state measurements $\mathbf{s}_t$, as illustrated in Figure 2. Three families of decision support systems were developed and evaluated:

1) optimization based on a predictive model, denoted as MPO (Model Prediction & Optimization),
2) recommender systems based on heterogeneous graph neural networks, denoted as HET,
3) hybrid decision support systems, denoted as HYB.

The proposed approaches were evaluated using 100 new simulation runs, all initialized with identical conditions (i.e., the same 100 random seeds). For comparison, each experiment was also repeated using fixed actions prescribed by the baseline recipe.

A. MPO decision support systems

Decision support systems based on *model-prediction optimization* (MPO) use the following components to recommend actions $\mathbf{a}_t$:

1) **Prediction model** M_t: Given the current state variables $\mathbf{s}_t$ and manipulative variables $\mathbf{a}_t$, the prediction model estimates the change in product mass in the next time step $y_{increase,t+24}$:

$$M_t(\mathbf{s}_t, \mathbf{a}_t) = y_{increase,t+24}. \tag{4}$$

The models considered were *XGBoost* (XGB) and a *multilayer perceptron* (MLP).

2) **Optimization method** Opt_t: Using the simulated measurements $\mathbf{s}_t$ together with the prediction model M_t, the optimization method determines the optimal action values $\mathbf{a}_t$:

$$Opt_t(\mathbf{s}_t, M_t) = \mathbf{a}_t^{opt}. \tag{5}$$

Here, the combination of $\mathbf{s}_t$ and M_t defines the objective function used to identify the optimal manipulative variables. The optimization techniques employed were *gradient descent* (GD), *Adam optimizer* (Adam), *Bayesian optimization* (BAY), *covariance matrix adaptation evolution strategy* (CMA-ES), and *differential evolution* (DE).

At each decision point, the system produced a recommended action $\mathbf{a}_t^{opt}$, which was subsequently applied in the simulation. Each MPO decision support system is denoted by the pattern *M + O*, where *M* refers to the prediction model (XGB or MLP), and *O* specifies the optimization method (e.g., *XGB + GD* indicates a system combining *XGBoost* with *gradient descent*).

B. HET decision support systems

Approaches based on *GNN-based recommender systems* use a graph-structured representation of the data to identify suitable manipulative actions for new experiments. The effectiveness of this approach depends largely on the choice of an appropriate graph representation and a suitable model architecture. These factors were examined in detail by You et al. [10], and their methodology was adapted and applied to determine the optimal parameters for our use case.

1) Graph Structure: The experimental data are modelled as a bipartite graph, as illustrated in Figure 3. The left set of nodes corresponds to the state-variable vectors $\mathbf{s}_{i,t}$, that is, the measurements obtained from past experiments at time t. The right set of nodes corresponds to the manipulative variables that the operator can adjust; these nodes do not contain additional attributes.

The connections between the nodes in both groups are weighted. Each edge between a state node $\mathbf{s}_{i,t}$ and a manipulative-variable node $v \in \{F_{oil}, F_{PAA}, F_s, F_g, F_w, T\}$ carries a weight corresponding to the value of the manipulative variable v used in experiment i at time t.

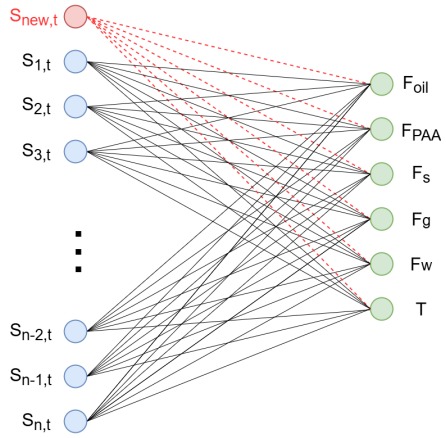

Fig. 3. Basic graph structure employed in the HET family of GNN-based recommender systems.

Two subfamilies of GNN models were developed: $HETv1$ and $HETv2$.

- **HETv1:** Models in this group were trained on graphs in which the state nodes $\mathbf{s}_{i,t}$ include, as an attribute, the observed change in penicillin mass $y_{increase,t+24}$. When actions for new nodes $\mathbf{s}_{new,t}$ were predicted and $y_{increase,t+24}$ was unknown, a high placeholder value (e.g., the 97th percentile of all known values) was assigned to focus the model's attention on past high-performing experiments. The percentile of the assigned placeholder value is also indicated in the model name (e.g., HETv1-97 or HETv1-3).

- **HETv2:** Models in this subfamily do not have access to the value $y_{increase,t+24}$. Instead, at each training step, the fraction of cases with the lowest penicillin-mass increase was removed and only the top $p\%$ of experiments was retained. If the top $p\%$ of samples was kept after filtering, this was indicated in the model name by the suffix *HETv2-Tp* (e.g., HETv2-T95 for a model that retains the top 95% of samples).

2) Exploration of GNN Design Space: In accordance with the methodology presented in [10], the space of possible GNN configurations was investigated . Our goal was to find the best architectures for predicting connection weights. Our hyperparameter space was defined using the following dimensions:

- **Activation function** $Act \in \{\text{ReLU}, \text{LeakyReLU}, \text{PreReLU}, \text{Swish}\}$
- **Aggregation mode** $Agg \in \{\max, \text{mean}, \text{sum}\}$
- **Connection threshold** $\varepsilon \in \{0.20, 0.15, 0.10, 0.05, 0.00\}$
- **Hidden-dimension size** $HidDim \in \{32, 64, 128\}$
- **Layer connectivity** $LayConn \in \{\text{skipcat}, \text{skipsum}, \emptyset\}$
- **Number of message-passing layers** $MPL \in \{2, 3, 4\}$
- **Number of post-processing layers** $PostPL \in \{0, 1, 2\}$
- **Number of preprocessing layers** $PrePL \in \{0, 1, 2\}$

The parameter connection threshold ε determines the degree of connectivity between user nodes. A connection between two user nodes is established, if the difference in the growth of the mass of penicillin between them is less than the *threshold* ε (values are normalized).

To analyse the effect of each design dimension, *the one-dimension-at-a-time exploration strategy* was adopted [10]. For every dimension under investigation, a set of random baseline architectures was first generated by fixing all remaining hyperparameters. Each candidate value of the analysed dimension was then inserted into every baseline architecture, producing a series of GNN variants that differed only in that single setting. All variants were trained and evaluated using the same train–validation split, and their performance was ranked according to the achieved validation error. This process was repeated across 100 random seeds, and the resulting rankings were aggregated to identify which values within each dimension consistently yielded the best performance.

Table II presents the optimal GNN model configurations for each time point of the experiment for the HETv1-97 model. HETv1-97 uses the time-specific configuration listed in Table II at every decision point and the suffix "97" indicates that the placeholder value of $y_{increase,t+24}$ assigned to a new node was set to the 97^{th} percentile of all known values. Several other models were also developed using suboptimal model hyperparameters and architectures, where the name extension indicates the deviation from the hyperparameters shown in Table II. All GNN models in the HETv2 family were trained using the same configurations as the HETv1-97 model.

3) HYB decision support systems: Table III lists the developed models from the *MPO* family and from the *HETv1* and *HETv2* subfamilies. These models were used to construct the *hybrid decision support systems*. Models from different families may exhibit varying performance depending on the decision time t. Thus, the hybrid models aim to exploit the complementary strengths of *MPO* and *HET* systems to achieve higher robustness and yield.

TABLE II
OVERVIEW OF SELECTED HYPERPARAMETERS BY TIME POINTS

	96 h	120 h	144 h	168 h	192 h	216 h
Act	PreReLU	LeakyReLU	LeakyReLU	Swish	Swish	LeakyReLU
Agg	max	max	max	max	sum	max
ϵ	*0.0*	*0.0*	*0.0*	*0.0*	*0.0*	*0.0*
HidDim	128	64	128	64	64	64
LayConn	skipsum	skipsum	skipsum	skipsum	skipsum	skipsum
MPL	2	2	2	2	2	2
PostPL	0	0	1	0	1	1
PrePL	0	0	0	0	0	0

The following hybrid models were defined:

- **HYBideal**: At each decision point, this model identifies the best-performing system among those listed in Table III and executes its recommended action. It serves as an *ideal* upper benchmark and is also used to generate an additional set of 100 training samples for the two subsequent hybrid models.
- **HYBv1**: Implemented as a simple MLP classifier that acts as a router, deciding which of the available decision support systems should be applied at decision time t. The routing decision is based on the current experimental state and on all recommended operator actions.
- **HYBv2**: Implemented as a simple MLP regressor that directly predicts new operator actions. Its recommendations are based on the state variables and on the actions suggested by the other decision support systems.

TABLE III
LIST OF DEVELOPED DECISION SUPPORT SYSTEMS GROUPED BY INDIVIDUAL FAMILIES.

MPO	**HETv1**	**HETv2**
XGB+BAY	HETv1-97	HETv2-T100
XGB+CMA-ES	HETv1-3	HETv2-T95
XGB+DE	HETv1-agg-97	HETv2-T90
MLP+BAY	HETv1-basic-97	HETv2-T85
MLP+CMA-ES	HETv1-basic-agg-97	HETv2-T80
MLP+DE		
MLP+Adam		
MLP+GD		

IV. RESULTS

The results of 100 test experiments were assessed according to three key criteria, each providing a different perspective on the proposed approaches.

A. Yield improvement and process stability

Table IV summarizes the results of 100 test experiments. In nearly all cases, the proposed decision support systems enabled better performance than simply following the predetermined recipe.

TABLE IV

VALIDATION RESULTS OF ALL MODELS ACROSS SEVERAL METRICS. VALUES ARE IN KILOGRAMS EXCEPT FOR THE LAST TWO COLUMNS. COLUMN **>3100** SHOWS HOW MANY EXPERIMENTS EXCEEDED 3100 KG, WHILE **IMP** DENOTES HOW MANY EXPERIMENTS IMPROVED COMPARED TO THE RECIPE.

Approach	Mean	SD	Min	Max	>3100	Imp
MLP+Adam	3367	564	**1599**	4171	80	**98**
MLP+GD	**4329**	1005	1487	**5361**	85	91
MLP+CMA-ES	3460	**540**	1438	4435	73	84
XGB+CMA-ES	3316	621	1506	4150	74	77
MLP+DE	4180	904	1549	5228	**87**	91
XGB+DE	3783	803	1369	5090	**87**	83
MLP+BAY	3873	1115	486	5074	74	81
XGB+BAY	3259	1130	0	5140	59	63
HETv1-97	**3948**	369	2329	**4522**	94	99
HETv1-agg-97	2653	684	1452	4729	14	20
HETv1-basic-97	3586	388	1794	4055	93	95
HETv1-basic-agg-97	3779	469	1890	4392	94	**100**
HETv1-3	3926	**301**	**2762**	4429	**97**	97
HETv2-T100	2797	661	1557	4733	20	30
HETv2-T95	3691	771	1556	4785	77	90
HETv2-T90	3917	**262**	**2990**	4508	**99**	**100**
HETv2-T85	**3986**	436	1417	**4964**	98	94
HETv2-T80	2577	518	1574	4109	14	14
HYBv1	**4642**	645	1633	**5431**	96	98
HYBv2	4566	**537**	**1676**	5077	**97**	**100**
HYBideal	*4906*	*385*	*1959*	*5499*	*99*	*100*
Recipe	*3007*	*650*	*1578*	*4044*	*47*	*/*

Among all decision support systems, the hybrid methods stand out due to their strong overall performance. Most models from the other families substantially improve the final penicillin yield, but they do not reach the performance achieved by the reference model *HYBideal* (average of 4906 kg). In contrast, the *HYBv1* (Figure 4) and *HYBv2* models come close to this benchmark, with average yields of 4642 kg and 4566 kg, respectively. Both models consistently exceed the prescribed performance threshold of 3100 kg in 95–100% of test cases. Moreover, their relatively low standard deviations indicate that these approaches are stable and less sensitive to variations in the data. The idealized *HYBideal* model achieves an even higher average (4906 kg), but this value should be interpreted as an estimate of the upper bound on achievable performance.

HYBv1 performs competitively with the best methods reported in the literature (Table V). It surpasses most previous approaches and achieves the highest observed yield (5431 kg), while HETv1 shows higher variability due to less frequent interventions. Prior work indicates that both overly frequent and overly sparse actions can hinder performance, making action frequency a key design choice. As our focus is operator support, the intervention rate reflects realistic practice. Overall, HYBv1 proves robust and aligns with the strongest previously published results for IndPenSim.

B. Nemenyi test

Based on the results presented in Table IV, an additional statistical analysis of the $k = 10$ models was conducted. Following the methodology of [11], each of the ten models was ranked according to final penicillin yield on $N = 100$ test experiments, from which average ranks were computed

TABLE V

COMPARISON OF DIFFERENT CONTROL STRATEGIES FOR THE PENICILLIN PRODUCTION PROCESS IN INDPENSIM. ALL EXPERIMENTS LASTED 230 HOURS. THE LAST COLUMN INDICATES THE FREQUENCY OF OPERATOR INTERVENTIONS.

Method	Ref.	Mean Yield $[kg]$	Range $[kg]$	Runs	Freq.
operator	[14]	2882 ± 745	∼1500–4200	30	irregular
SBC	[14]	2912 ± 786	∼1600–4300	30	fixed profile
PI control	[14]	3517 ± 315	∼3000–4000	30	every 12 min
NMPC	[13]	3987 ± 265	∼3500–4400	30	every 12 min
SAC	[13]	3944 ± 272	∼3450–4400	30	every 12 min
10-bit RPC	[12]	4864 ± 318	∼4400–5200	50	12 min / 1 h / 2 h
HYBv1		4642 ± 645	1633–5431	100	24 h

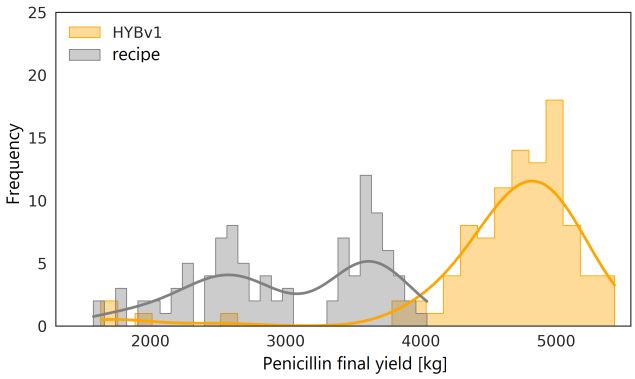

Fig. 4. Distribution of the final mass of penicillin using HYBv1 recommendations and without them.

(Table VI). *HYBideal* achieved the best average rank (1.21), followed by the remaining hybrid variants.

Friedman's test ($\chi_F^2 = 594.07$, $p \approx 3.9 \times 10^{-122}$) and the Iman–Davenport correction ($F_F = 192.24$, $p \approx 1.1 \times 10^{-16}$) reject the null hypothesis of equal model performance, supporting post-hoc analysis. Using the Nemenyi test, two models are considered statistically indistinguishable when their average ranks differ by less than the critical difference

$$\mathrm{CD}_{0.05} = q_{0.95}(k, \infty)\sqrt{\frac{k(k+1)}{6N}} = 1.355,$$

as illustrated in Figure 5.

TABLE VI

MEAN RANKS AND STANDARD DEVIATIONS OF THE EVALUATED MODELS OVER 100 EXPERIMENTS (LOWER VALUES INDICATE BETTER PERFORMANCE).

Model	Mean Rank	SD
HYBideal	1.21	0.682
HYBv1	2.92	1.914
HYBv2	3.99	1.467
MLP+GD	4.29	2.418
MLP+DE	5.29	2.222
MLP+BAY	6.15	2.245
HETv1-97	7.00	1.146
HETv2-T85	7.05	1.684
HETv2-T90	7.57	1.266
recipe	9.53	0.688

Overall, the support systems substantially outperform the baseline prescription approach. Hybrid methods achieve the best results; HYBv1 and HYBv2 do not differ significantly,

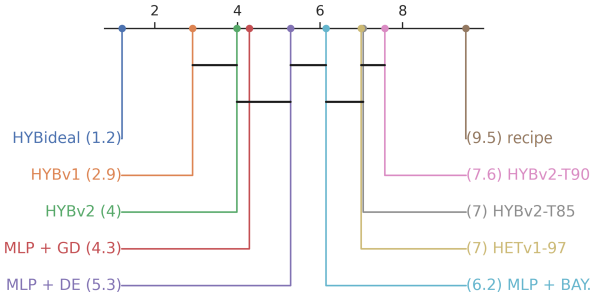

Fig. 5. The critical difference diagram shows which models are not statistically different at $\alpha = 0.05$. Models with rank differences smaller than the CD are connected by horizontal lines.

while MLP+GD, although effective, performs significantly worse than HYBv1.

C. Variation in Model Recommendations

Across 100 new batch experiments, the recommended corrective actions from different support systems were compared to evaluate how much their suggestions differed. The analysis focused on the MPO, HETv1, and HETv2 families—the foundations of the hybrid HYB models. For each model and time t, all prior recommendations were concatenated into an action vector, and similarities between models were computed, yielding the dendrogram in Figure 6. The clear differences between model families motivate the hybrid approach: by combining complementary behaviours, HYB models produce more consistent recommendations and achieve superior overall performance.

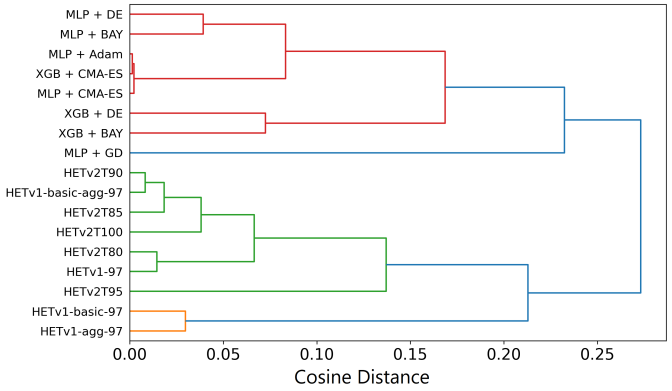

Fig. 6. Cosine distances between model recommendation vectors at $t = 96\ h$.

V. Conclusion

The paper evaluated several decision support approaches for improving penicillin production in the *IndPenSim* environment. Three families of methods were examined—model-based optimisation, GNN-based recommenders, and hybrid approaches—each generating corrective actions from historical process data.

The main conclusions are as follows:

1) All proposed decision support systems significantly improve the final penicillin yield compared with recipe-based operator control.
2) Models within the same family tend to recommend similar corrective actions, and hybrid approaches benefit from combining these complementary behaviours. As a result, the hybrid models achieve the most stable and highest overall performance.
3) Compared with previously published solutions, the proposed systems achieve competitive results while requiring substantially fewer interventions, highlighting their suitability for operator-assistance scenarios rather than fully autonomous control.

Acknowledgment

This work was supported by *DIGITOP, GA no. TN-06-0106*, funded by *Ministry of Higher Education, Science and Innovation of Slovenia, Slovenian Research and Innovation Agency, and European Union - NextGenerationEU,* and by *Slovenian Research Agency* grant *P2-0001*.

References

[1] I. Bucci, V. Fani, and R. Bandinelli, "Towards human-centric manufacturing: exploring the role of human digital twins in Industry 5.0," *Sustainability*, vol. 17, no. 1, article 129, 2025.
[2] S. French, "Decision support systems," in *e-Democracy: A Group Decision and Negotiation Perspective*, D. Rios Insua and S. French, Eds. Dordrecht: Springer Netherlands, 2010, pp. 65–82. doi: 10.1007/978-90-481-9045-4_5.
[3] D. J. Power, "Understanding data-driven decision support systems," *Information Systems Management*, vol. 25, no. 2, pp. 149–154, 2008.
[4] P. Lacomme, G. Rault, and M. Sevaux, "Integrated decision support system for rich vehicle routing problems," *Expert Systems with Applications*, vol. 178, p. 114998, 2021.
[5] W. K. Wai, L. Ting, L. W. Pang, H. Budihardjo, G. C. Liang, D. Liya, F. Zhu, and C. P. T-Howe, "Decision support system for production scheduling (DSSPS)," *Procedia Computer Science*, vol. 96, pp. 315–323, 2016.
[6] S. Taheri Moghadam *et al.*, "The effects of clinical decision support system for prescribing medication on patient outcomes and physician practice performance: a systematic review and meta-analysis," *BMC Medical Informatics and Decision Making*, vol. 21, no. 1, p. 98, 2021.
[7] A. Thelen, X. Zhang, O. Fink, Y. Lu, S. Ghosh, B. D. Youn, M. D. Todd, S. Mahadevan, C. Hu, and Z. Hu, "A comprehensive review of digital twin—part 1: modeling and twinning enabling technologies," *Structural and Multidisciplinary Optimization*, vol. 65, no. 12, p. 354, 2022.
[8] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: a survey," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–37, 2022.
[9] S. Goldrick, A. Ştefan, D. Lovett, G. Montague, and B. Lennox, "The development of an industrial-scale fed-batch fermentation simulation," *Journal of Biotechnology*, vol. 193, pp. 70–82, 2015. Elsevier.
[10] J. You, Z. Ying, and J. Leskovec, "Design space for graph neural networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17009–17021, 2020.
[11] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. 1, pp. 1–30, 2006.
[12] G. Simethy, M. Bauer, R. Tan, and F. Buelow, "Robust Predictable Control (RPC) for Optimizing Fed-Batch Penicillin Production," *IFAC-PapersOnLine*, vol. 59, no. 6, pp. 385–390, 2025.
[13] H. Li, T. Qiu, and F. You, "AI-based optimal control of fed-batch biopharmaceutical process leveraging deep reinforcement learning," *Chemical Engineering Science*, vol. 292, p. 119990, 2024.
[14] S. Goldrick, C. A. Duran-Villalobos, K. Jankauskas, D. Lovett, S. S. Farid, and B. Lennox, "Modern day monitoring and control challenges outlined on an industrial-scale benchmark fermentation process," *Computers & Chemical Engineering*, vol. 130, p. 106471, 2019.