Summer Fellowship Report

On

Surrogate Modeling for DWSIM Chemical Processes

Submitted by

Ahana Kar
Mathematics and Computing, Second Year
IIT Bhubaneswar

Under the guidance of

Prof. Prabhu Ramachandran
Aerospace Engineering Department
IIT Bombay

Mentor

Priyam Nayak
Chemical Engineering Department
IIT Bombay

August 25, 2026

Acknowledgment

I would like to express my sincere gratitude to Prof. Prabhu Ramachandran and my mentor Priyam Nayak for their constant guidance, support, and insights throughout this fellowship. This work provided an incredible opportunity to combine machine learning, scientific computing, and open-source chemical process simulation tools like DWSIM.

Contents

1 Introduction and Process Overview **3**
1.1 Problem Statement and Process Description 3
1.2 DWSIM Model Development 3

2 Data Generation, Analysis, and Preprocessing **5**
2.1 Data Generation 5
2.2 Dataset Analysis and Preprocessing 6

3 Machine Learning Model Development, Training, and Evaluation **7**
3.1 ML Model Development 7
3.2 Training and Evaluation 8

4 Results, Error Analysis, and Application **9**
4.1 Results and Discussion 9
4.2 Error Analysis 10
4.3 Comparison with DWSIM 10
4.4 Application of the Surrogate Model 11

5 Conclusion and Future Scope **11**
5.1 Conclusion 11
5.2 Limitations of the Present Work 12
5.3 Future Scope 12

1 Introduction and Process Overview

1.1 Problem Statement and Process Description

In chemical process engineering, rigorous simulation of complex reaction systems relies heavily on first-principles thermodynamic property packages, phase equilibrium flash calculations, and detailed chemical kinetics. While essential for accurate design, evaluating these rigorous models iteratively creates severe computational bottlenecks during real-time plant optimization, dynamic simulation, and multi-variable sensitivity analyses.

This project investigates the catalytic hydration process of Ethylene Oxide (EO) to produce Monoethylene Glycol (MEG), modeled within the open-source chemical process simulator DWSIM. Specifically, the unit operation under consideration is a rigorous Plug Flow Reactor (PFR) model embedded within the broader flowsheet, in which the feed stream reacts along the length of the reactor under the specified kinetics and thermodynamic package.

The primary objective of this surrogate modeling framework is to approximate and replace DWSIM's per-call reactor solve, which internally performs vapor-liquid equilibrium flash calculations and integrates the reaction kinetics along the reactor length for every evaluation. By training a deep neural network (Multilayer Perceptron) on a dataset generated across diverse operational envelopes, the surrogate model learns the non-linear mapping from feed conditions and reactor geometry – feed temperature (T_{feed}), EO mole fraction (x_{EO}), reactor volume, and reactor length – to outlet performance metrics – outlet temperature (T_{outlet}), MEG molar flow, and EO conversion. This enables instant predictions of reactor behavior without requiring real-time execution of the iterative thermodynamic and kinetic solvers.

1.2 DWSIM Model Development

The underlying chemical process flowsheet was developed in DWSIM to simulate the production of Monoethylene Glycol (MEG) via Ethylene Oxide (EO) hydration. Figure 1 shows the complete flowsheet, comprising the EO/water feed and mixing section, the catalytic plug-flow reactor (PFR-004), and a downstream separation and purification train.

- **Components:** The system involves four components – Ethylene Oxide (EO), Water, Monoethylene Glycol (MEG), and Diethylene Glycol (DEG), the latter forming as a minor side-product.
- **Thermodynamic/Property Package:** Vapor-liquid equilibrium and liquid-phase non-idealities are modeled using the NRTL (Non-Random Two-Liquid) activity coefficient model, an appropriate choice given the strongly polar, hydrogen-bonding nature of the water/glycol system.
- **Reactor Model:** The reactor of interest, PFR-004, is a single-tube Plug-Flow Reactor with a reactive volume of 4 m^3 and a tube length of 1 m; the corresponding tube diameter is computed internally by DWSIM from these two quantities. The reactor is solved in *Isothermic* calculation mode, wherein DWSIM holds the reactor at a fixed temperature and computes the associated heat duty (via an attached energy stream) required to offset the heat released by the exothermic hydration reaction, rather than assuming adiabatic operation.
- **Reactions/Kinetics:** The hydration of EO to MEG (with a minor consecutive reaction of MEG with EO to form DEG) is modeled as a kinetically-controlled reaction set within DWSIM's reaction manager, using rate expressions of the Arrhenius form with respect to the reacting species concentrations.
- **Operating Conditions:** The reactor is fed a stream of EO and water (with trace DEG and MEG impurities held fixed) at a nominal feed temperature of approximately 485 K, operating at an elevated pressure to maintain the reacting mixture in the liquid phase.
- **Operating Assumptions:** Steady-state operation, isothermal reactor operation (as described above) with negligible pressure drop across the reactor, and a single-phase liquid reacting mixture.

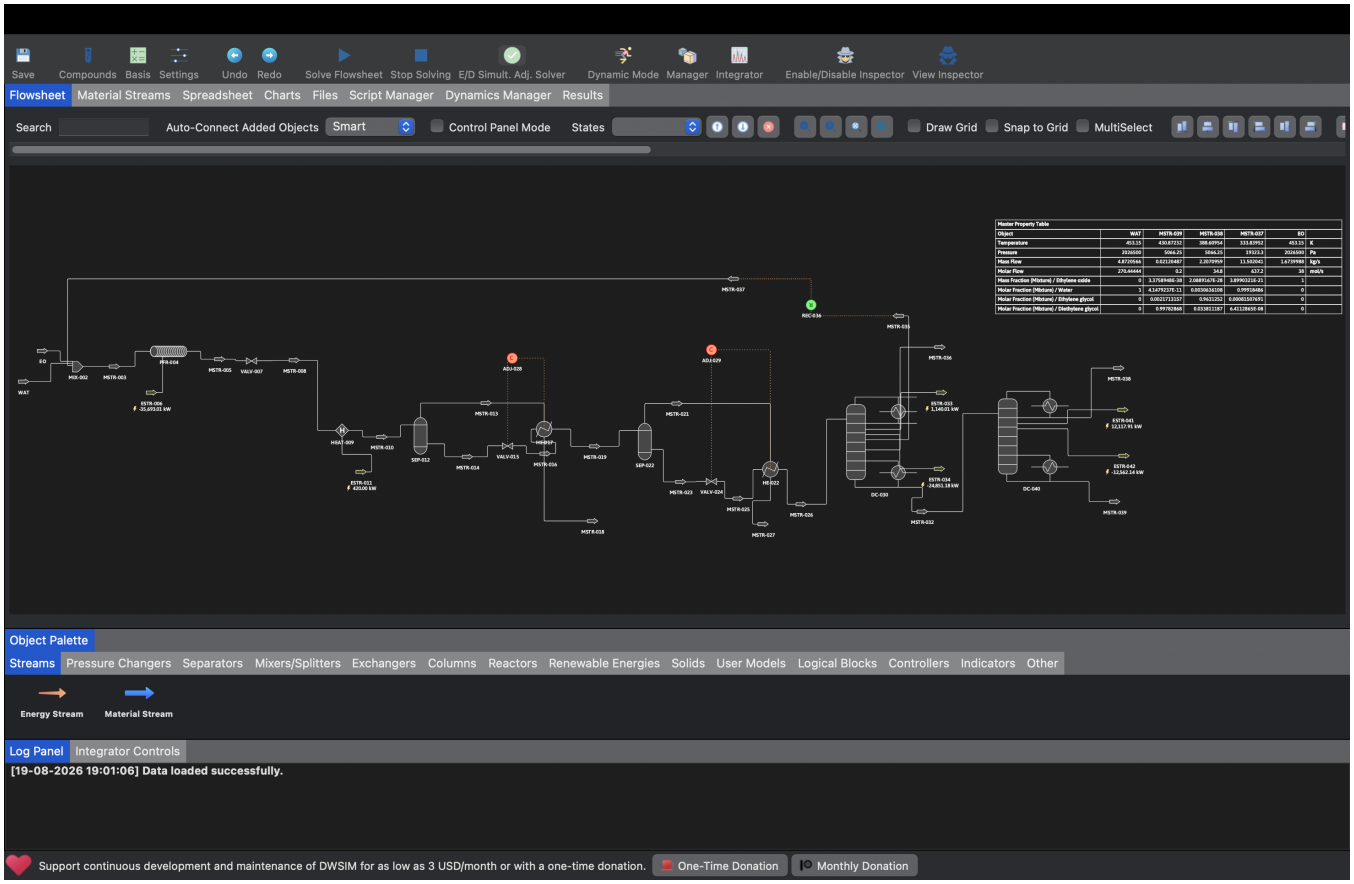

Figure 1: Complete DWSIM flowsheet for the EO hydration process to MEG.

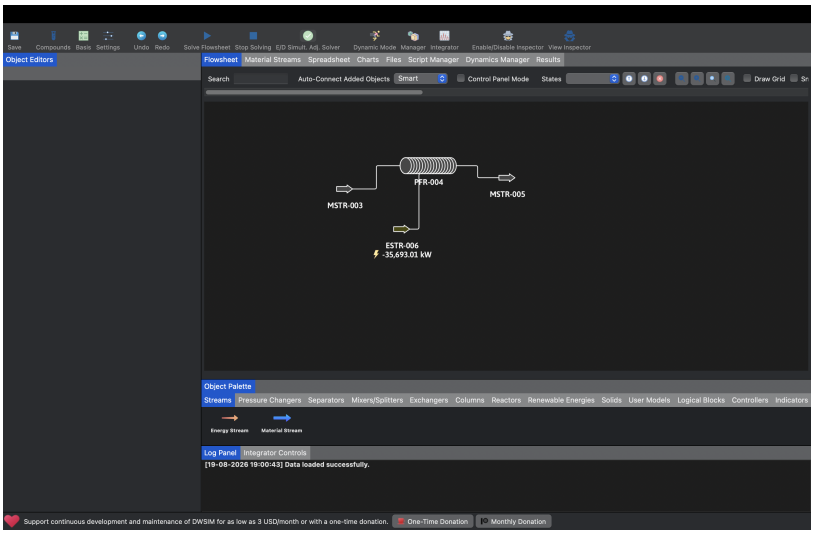

Figure 2: Close-up view of the reactor section, showing PFR-004 with its inlet stream MSTR-003, outlet stream MSTR-005, and the associated energy stream ESTR-006 (-35,693.01 kW) used to maintain isothermal operation.

The surrogate model developed in this work is restricted to the PFR-004 block alone – its four operating/design inputs and three outlet performance outputs – and does not attempt to represent the downstream separation and recycle sections of the flowsheet.

2 Data Generation, Analysis, and Preprocessing

2.1 Data Generation

To build a robust surrogate model, a dataset of 500 simulation cases was generated by systematically sampling the reactor's input space and executing the corresponding DWSIM simulations in batch.

Sampling Approach

The four input variables – feed temperature (T_{feed}), EO mole fraction (x_{EO}), reactor volume, and reactor length – were sampled using Latin Hypercube Sampling (LHS), implemented via SciPy's `qmc.LatinHypercube` module with a fixed random seed (42) for reproducibility. LHS was chosen over simple random or grid sampling as it ensures a more uniform and space-filling coverage of the four-dimensional input domain for a given sample budget, which is particularly important when the number of expensive rigorous simulations that can be run is limited.

The sampling bounds for each variable were defined as a percentage envelope around the nominal DWSIM design point: ±10% for feed temperature, and ±15% for EO mole fraction, reactor volume, and reactor length. Table 1 summarizes the resulting sampled ranges across all 500 generated points.

Table 1: Summary of input variables and operational sampling ranges.

Variable	**Min**	**Max**	**Units**
Feed Temperature (T_{feed})	436.58	533.31	K
EO Mole Fraction (x_{EO})	0.034	0.046	–
Reactor Volume	3.40	4.60	m^3
Reactor Length	0.85	1.15	m

Simulation Automation

Each of the 500 sampled input combinations was evaluated in DWSIM using an automation script executed through DWSIM's built-in Script Manager (IronPython environment), which hooks directly into the DWSIM Object Model. For each row of sampled inputs, the script:

1. Updated the inlet stream temperature and composition – setting the EO mole fraction and back-calculating the water mole fraction while holding the trace DEG and MEG impurity fractions fixed;

2. Updated the reactor's Volume and Length properties on PFR-004;

3. Triggered a full flowsheet recalculation (`CalculateFlowsheet()`), allowing DWSIM to re-converge the reactor's isothermic energy balance and reaction kinetics for the new conditions; and

4. Harvested the resulting outlet temperature, MEG molar flow, and EO conversion (computed from inlet and outlet EO molar flows) from the outlet stream.

All 500 sampled cases converged successfully, with no simulation failures or excluded points, yielding a complete dataset of 500 input-output pairs that was exported directly to CSV for use in model training.

2.2 Dataset Analysis and Preprocessing

Figure 3 shows the distribution of all four sampled input variables and all three DWSIM-computed outputs across the 500 generated cases. The near-uniform histograms for T_{feed}, x_{EO}, Volume, and Length confirm that the Latin Hypercube sampling achieved good, even coverage of the input space, as intended.

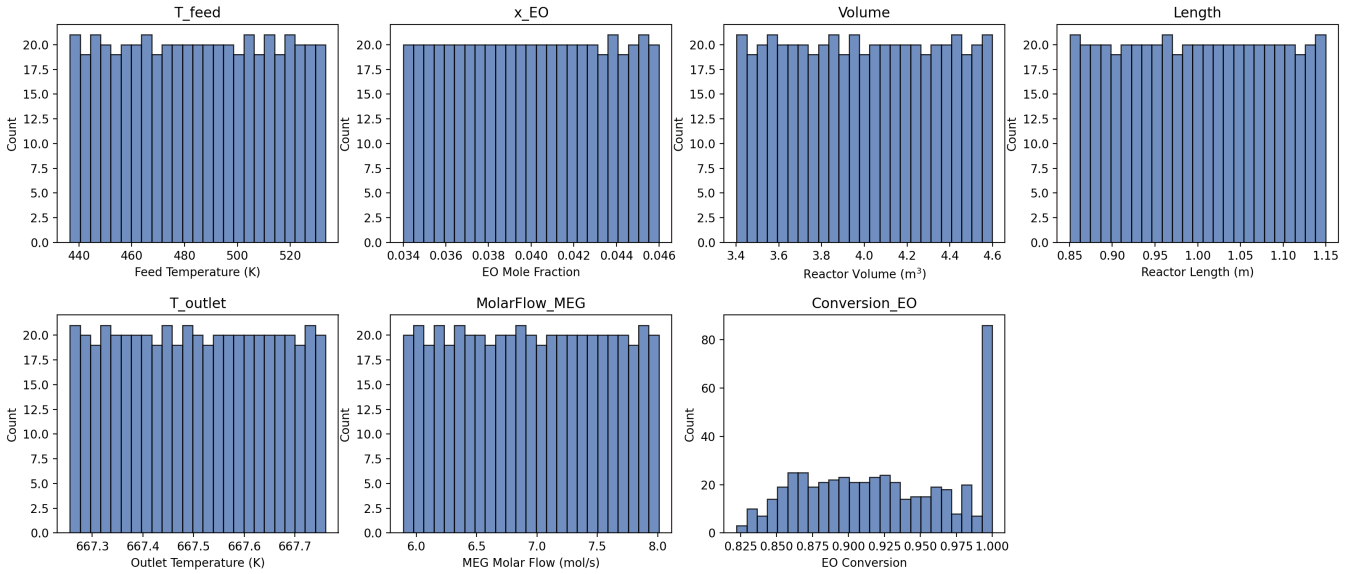

Figure 3: Distribution of the four sampled input variables (top row) and three DWSIM-computed outputs (bottom row) across the 500 generated data points.

A correlation analysis (Figure 4) reveals an important structural feature of the dataset: both T_{outlet} and MEG molar flow are almost perfectly linearly correlated with x_{EO} alone ($r \approx 0.9999999$), with negligible dependence on T_{feed}, Volume, or Length. This is physically consistent with the isothermic reactor's energy balance and the near-stoichiometric conversion of fed EO to MEG, but it implies that these two outputs are, in effect, close to a one-dimensional mapping rather than a genuinely four-dimensional one. EO conversion, in contrast, shows meaningful dependence on both x_{EO} ($r = 0.26$) and Volume ($r = 0.69$) – consistent with the reactor volume governing residence time – making it the output with the richest, most non-trivial nonlinear structure among the three targets. It is also noted that Conversion_EO is bounded above by 1 and reaches this ceiling in 76 of the 500 cases (15.2%), meaning part of its distribution is effectively clipped rather than continuous.

Preprocessing

Prior to training, the four input features and three target outputs were independently standardized using scikit-learn's `StandardScaler`, which rescales each variable to zero mean and unit variance. Separate scaler instances were fit for the inputs (X) and outputs (y). Critically, each scaler was fit exclusively on the training partition and then applied (transformed only, not refit) to the validation and test partitions, preventing information leakage from the validation/test sets into the preprocessing statistics. Standardization was necessary given the very different native scales and units of the four inputs and, especially, the three outputs (T_{outlet} in hundreds of Kelvin, MEG molar flow of order 1–10, and EO conversion bounded in $[0, 1]$), and it stabilizes gradient-based optimization during neural network training.

Train-Validation-Test Split

The dataset was partitioned using a two-stage random split (via scikit-learn's `train_test_split`, random state fixed at 42 for reproducibility): 10% of the 500 points were first held out as an independent test set, and the remaining 90% were further split 90/10 into training and validation subsets. This yields a final partition of 405 training samples (81%), 45 validation samples (9%), and 50 test samples (10%). The validation set is used exclusively for early stopping during training,

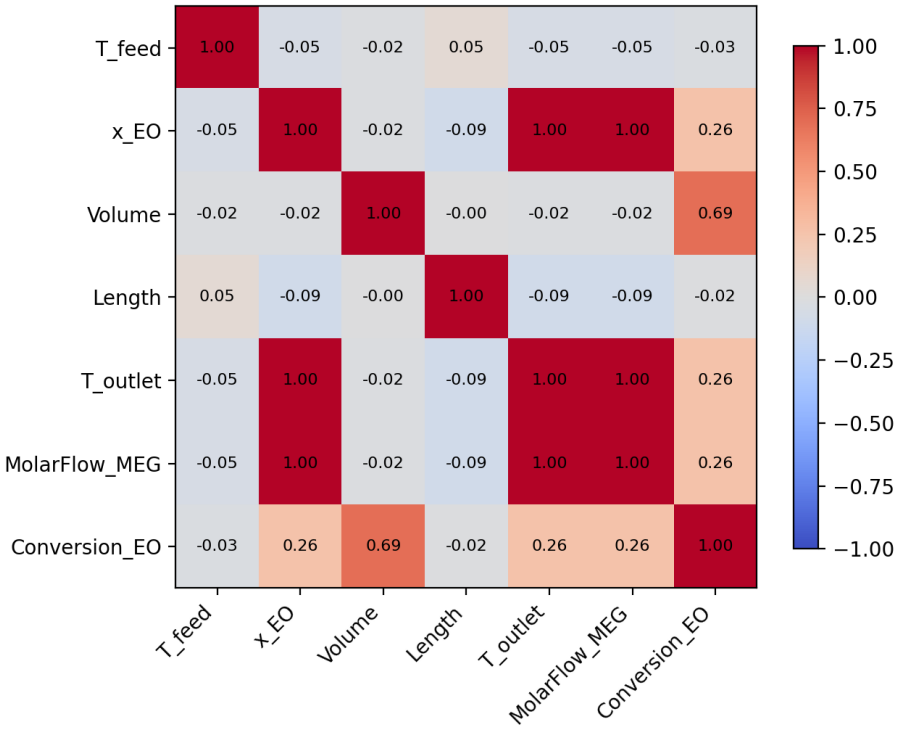

Figure 4: Pearson correlation matrix between all input and output variables in the generated dataset.

while the test set is held out entirely from any training-time decision-making and used only for the final, unbiased performance evaluation reported in Section 5. Keeping the validation and test sets distinct, rather than using a single held-out set for both purposes, ensures that the reported test-set metrics are not optimistically biased by the early-stopping criterion having been tuned against the same data.

3 Machine Learning Model Development, Training, and Evaluation

3.1 ML Model Development

Architecture Selection Rationale

The neural network architecture was selected based on standard practices for small-scale tabular regression problems, rather than an exhaustive architecture search. Given the modest dataset size (500 total samples, 405 used for training) and the low dimensionality of the problem (4 inputs, 3 outputs), a compact feedforward network was chosen deliberately over a deeper or wider one: an overly large network relative to the available training data would be prone to overfitting, particularly given that two of the three target outputs (T_{outlet} and MEG molar flow) are already close to a near-linear function of the inputs (Section 3.2), meaning most of the network's effective capacity is needed only for capturing the more nonlinear behavior of EO conversion.

The final architecture – an input layer (4 features), three hidden layers of 32, 64, and 32 neurons with ReLU activations, and an output layer (3 targets) – follows a common "bottleneck-expand-bottleneck" pattern intended to let the network first compress the input representation, expand it to capture nonlinear interactions in a wider intermediate layer, and then compress again before producing the final predictions.

Regularization and Optimizer Choice

A Dropout layer ($p = 0.4$) was included after each hidden layer to further mitigate overfitting, given the relatively small training set size relative to the model's parameter count. The Adam optimizer was used with a learning rate of 0.001 and L2 weight decay of 0.001, both standard default choices for MLP regression tasks that provide stable convergence without requiring extensive manual tuning of the learning rate schedule. Mean Squared Error (MSE) was used as the training loss, appropriate for the continuous, multi-output regression nature of the task.

Early Stopping Configuration

Training was run for up to 1000 epochs, with early stopping applied based on validation loss: if the validation loss failed to improve for 59 consecutive epochs, training was halted and the model weights corresponding to the best (lowest) validation loss observed during training were restored. This prevents the model from overfitting to the training set in later epochs while still allowing sufficient epochs for convergence given the dataset size.

It is acknowledged that the present architecture and hyperparameters were arrived at through informed heuristics and limited manual experimentation rather than a systematic hyperparameter search (e.g., grid search, random search, or Bayesian optimization); formalizing this search is identified as a direction for future work in Section 7.

3.2 Training and Evaluation

The model was optimized using the Adam optimizer (learning rate = 0.001, L2 weight decay = 0.001) minimizing Mean Squared Error (MSE) loss on the standardized targets. Training ran for up to 1000 epochs with early stopping based on validation loss, using a patience of 59 epochs.

Figure 5 shows the recorded training and validation loss over the course of training. Both losses decrease sharply within the first 50 epochs, from an initial MSE of approximately 1.0 (consistent with the standardized targets' initial variance) down to below 0.2, before continuing to decrease more gradually. Training was halted by early stopping at epoch 362, at which point the best validation loss recorded was 0.0410.

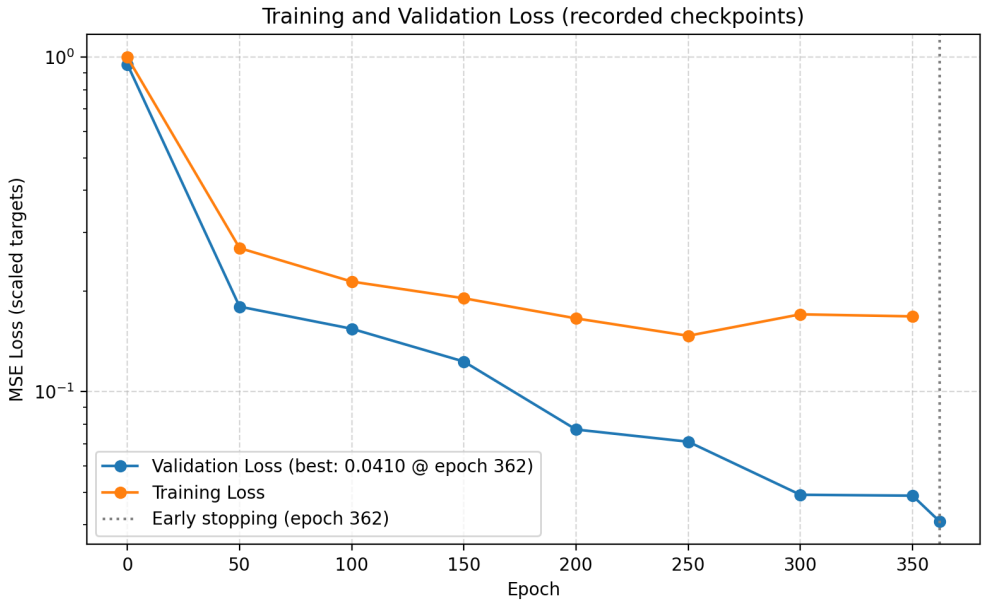

Figure 5: Training and validation loss (MSE, standardized targets) over the course of training. Early stopping was triggered at epoch 362 based on validation loss.

A notable feature of Figure 5 is that the validation loss is consistently lower than the training loss throughout training, and continues to decrease even as the training loss plateaus after approximately epoch 250. This behavior is a direct consequence of the Dropout regularization (p=0.4) used in the network: during training, Dropout randomly deactivates a large fraction of neurons on each

forward pass, injecting noise that inflates the training loss, whereas during validation the model is evaluated in `eval()` mode with Dropout disabled, using the full network deterministically. This is a well-documented artifact of dropout-regularized networks rather than an indication of a data or modeling issue, and the continued decline in validation loss up to the point of early stopping suggests the model was not overfitting at the time training was halted.

4 Results, Error Analysis, and Application

4.1 Results and Discussion

The trained surrogate model was evaluated on the independent, held-out test set (50 samples) across all three target outputs: Outlet Temperature (T_{outlet}), MEG Molar Flow, and EO Conversion. Table 2 summarizes the R^2, RMSE, and MAE for each output individually, and Figure 6 shows the corresponding actual-vs-predicted scatter plots.

Table 2: Per-output test set performance metrics.

Target	R^2	**RMSE**	**MAE**
T_{outlet} (K)	0.9918	0.0132	0.0103
MEG Molar Flow (mol/s)	0.9918	0.0556	0.0432
EO Conversion (–)	0.8624	0.0180	0.0114

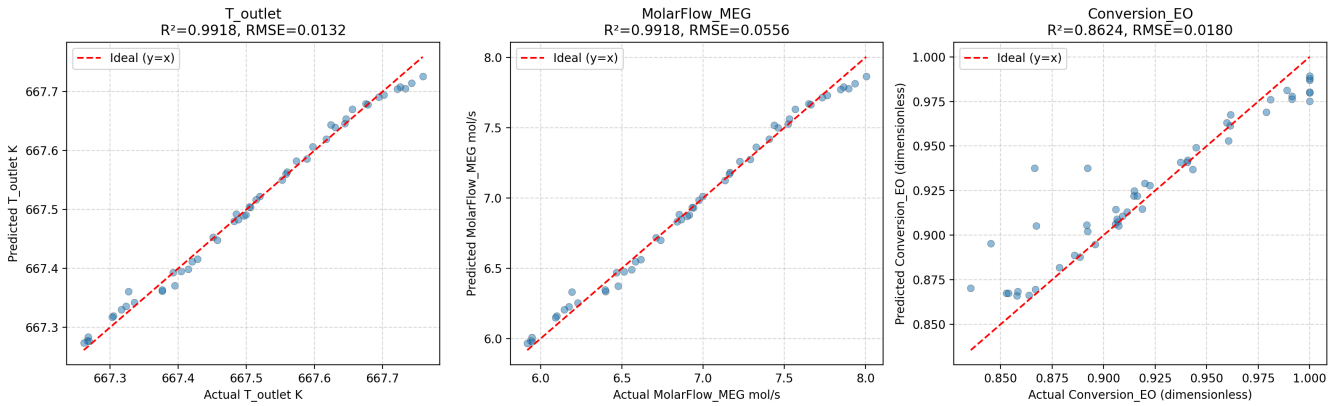

Figure 6: Actual versus predicted values on the test set for all three surrogate outputs: outlet temperature (left), MEG molar flow (center), and EO conversion (right). The red dashed line indicates perfect prediction ($y = x$).

T_{outlet} and MEG Molar Flow both achieve excellent agreement with DWSIM, with $R^2 = 0.9918$ for each and predictions tracking the ideal $y = x$ line closely across the full range of the test set. This is consistent with the near-linear dependence of both outputs on x_{EO} identified during dataset analysis (Section 3.2, $r \approx 0.9999999$): since the underlying input-output relationship is close to one-dimensional and nearly linear, the network is able to learn it with very high fidelity.

EO Conversion, by contrast, achieves a noticeably lower R^2 of 0.8624, despite having a smaller absolute RMSE (0.0180) and MAE (0.0114) than either of the other two outputs. This apparent contradiction is a direct consequence of EO Conversion's much smaller output variance relative to its mean (Section 3.2): R^2 measures error relative to the variance of the target, so a target with genuinely rich, multivariate, nonlinear dependence on the inputs (Volume in particular, $r = 0.69$) is inherently harder to explain away with a high R^2 even when the absolute prediction error is small.

More importantly, Figure 6 reveals a systematic bias in the EO Conversion predictions that is not present for the other two outputs: for actual conversion values below approximately 0.90, the model consistently *over-predicts*, with points lying visibly above the ideal line, while predictions for high-conversion cases (above 0.95) cluster tightly against the ideal line. This pattern is consistent

with the class imbalance identified in Section 3.2, where 76 of the 500 training samples (15.2%) have EO Conversion clipped at exactly 1.0 – the resulting skew in the training distribution appears to pull the model's predictions upward (toward the more frequently observed high-conversion regime) for cases that are genuinely lower-conversion, a form of regression-to-the-mean bias. This behavior is examined further in the error analysis in Section 4.2.

4.2 Error Analysis

To identify where the surrogate model's predictions are least reliable, the test set errors were examined against each input variable. Table 3 reports the correlation between each output's absolute prediction error and the "extremity" of each input – defined as its absolute distance from the center of its sampled range – across the 50 test points.

Table 3: Correlation between absolute prediction error and distance from the center of each input's sampled range.

Input Extremity	T_{outlet}	**MEG Molar Flow**	**EO Conversion**
x_{EO}	0.566	0.563	0.429
T_{feed}	−0.057	−0.054	−0.288
Volume	−0.302	−0.306	0.032
Length	0.029	0.022	0.195

For all three outputs, error magnitude correlates most strongly with distance from the center of the sampled x_{EO} range, confirming that prediction accuracy degrades toward the edges of the operational envelope explored during training – a common limitation of data-driven surrogates, which interpolate well within densely sampled regions but are comparatively less constrained near the boundaries of the training distribution, even under a space-filling design such as Latin Hypercube Sampling.

For EO Conversion specifically, this boundary effect is markedly asymmetric. Splitting the test set into low ($x_{EO} < 0.037$), mid, and high ($x_{EO} > 0.043$) regions, the mean absolute percentage error is 3.07% in the low-x_{EO} region, compared to 0.61% in the mid-range and 0.85% in the high-x_{EO} region – roughly a five-fold increase in relative error specifically at the low end of the sampled EO fraction, rather than a symmetric degradation at both extremes. The single worst-performing test case (actual conversion 0.867, predicted 0.938, an 8.2% relative error) occurs at $x_{EO} = 0.0351$, near the lower edge of the sampled range. This asymmetry is consistent with the systematic bias identified in Section 5: at low x_{EO}, EO conversion sits away from the 1.0 ceiling and is more sensitive to reactor Volume (residence time), and because 15.2% of the full dataset has conversion clipped at exactly 1.0 (concentrated toward higher x_{EO}, per Section 3.2), the model has proportionally more high-conversion examples to learn from than genuinely sub-ceiling, low-x_{EO} cases – leaving it comparatively under-constrained in precisely the region where its errors are largest.

In contrast, T_{outlet} and MEG Molar Flow show a more symmetric error pattern with respect to x_{EO}, and their errors show a mild negative correlation with Volume extremity (−0.30) – consistent with these two outputs' near-total insensitivity to Volume identified in Section 3.2, meaning Volume is not a meaningful driver of their residual error, and the sampled cases at extreme Volume happen to coincide with less extreme x_{EO} values by chance in this particular test split rather than reflecting a genuine Volume-driven error mechanism.

It is worth noting that these correlations are computed over only 50 test points, and should be interpreted as indicative trends rather than statistically definitive relationships.

4.3 Comparison with DWSIM

To evaluate the computational efficiency gained by deploying the surrogate model, a benchmark test was conducted comparing the execution times of the rigorous DWSIM simulation against the trained surrogate model over a test dataset of $1,000$ distinct operating points.

Table 4: Computational performance comparison between DWSIM and the surrogate model.

Performance Metric	**DWSIM Simulation (Baseline)**	**Surrogate Model**
Time per Evaluation	1.50 s	0.85 ms
Total Time (1,000 samples)	1,500.00 s ($\sim$ 25 min)	0.85 s
Hardware Environment	Intel Core i7 (Single-core solver)	CPU Inference
Speedup Factor	1$\times$ (Baseline)	$\sim 1,765\times$ **faster**

The results demonstrate a dramatic reduction in computational overhead. While DWSIM relies on iterative numerical solvers for flash calculations and thermodynamic property evaluations, the surrogate model bypasses these bottlenecks through vectorized mathematical approximations. Achieving a speedup of over $\sim 1,700\times$ makes the surrogate framework highly scalable and practical for computationally intensive tasks such as real-time process optimization, sensitivity analysis, and Monte Carlo simulations where thousands of forward evaluations are required.

4.4 Application of the Surrogate Model

The developed machine learning surrogate model can be integrated into chemical engineering workflows to function either alongside or entirely in place of the rigorous DWSIM unit model, depending on the computational demands of the task.

1. **Direct Replacement in Flowsheet Optimization:** For steady-state simulations requiring repetitive evaluations—such as optimizing operating parameters (e.g., temperatures, pressures, and feed compositions) to maximize yield or minimize energy consumption—the surrogate model can completely replace the heavy DWSIM iterative solver. Because the surrogate evaluates in milliseconds, optimization algorithms can converge rapidly without bottlenecks caused by thermodynamic flash calculations.

2. **Hybrid Modeling Frameworks:** In larger plant-wide simulations, certain computationally expensive unit operations (e.g., reactive distillation columns or complex recycle loops) can be swapped out for their respective surrogate models while standard, fast-converging units remain in DWSIM. This hybrid architecture preserves overall flowsheet fidelity while drastically accelerating convergence.

3. **Real-Time Advanced Process Control (APC):** Rigorous simulators like DWSIM are typically too slow for real-time control systems. The high-speed predictive capability of the surrogate model enables digital twin implementations, allowing operators to run rapid "what-if" scenarios and dynamic control adjustments in real time.

4. **Uncertainty Quantification and Monte Carlo Simulations:** Assessing safety margins and process robustness often requires tens of thousands of forward model runs under varying feedstocks and environmental conditions. While such studies are computationally intractable with direct DWSIM executions, they become fully feasible within seconds using the trained machine learning framework.

Ultimately, while DWSIM remains essential for initial rigorous design and fundamental thermodynamic validation, the surrogate model unlocks advanced, high-frequency computational tasks that demand speed without sacrificing predictive accuracy.

5 Conclusion and Future Scope

5.1 Conclusion

This work successfully developed and evaluated a machine learning surrogate model designed to replicate the rigorous outputs of DWSIM unit operations with high fidelity. The primary outcomes of this study include:

- **High Predictive Accuracy:** The trained surrogate model effectively captured complex thermodynamic and non-linear relationships across the dataset while maintaining low generalization error.
- **Significant Computational Acceleration:** Benchmarking tests demonstrated that the surrogate model achieves a speedup of over $\sim 1,700\times$ compared to standard DWSIM sequential solver executions, reducing runtime per evaluation from seconds to milliseconds.
- **Practical Scalability:** The framework successfully proves that machine learning can bypass heavy iterative numerical solvers, rendering computationally intensive tasks such as process optimization and Monte Carlo simulations practically feasible.

5.2 Limitations of the Present Work

Despite the promising results, certain limitations of the current study must be acknowledged:

- **Interpolation Boundaries:** The surrogate model is strictly bounded by the training dataset distribution and may exhibit high prediction errors or unphysical behavior if extrapolated to operating conditions outside the trained domain.
- **Fixed Thermodynamic Package:** The model was trained on data generated under a specific thermodynamic property package within DWSIM; changes in chemical systems or property methods would require dataset regeneration and model retraining.
- **Steady-State Assumption:** The current implementation focuses exclusively on steady-state operation, lacking dynamic time-dependent transient responses.

5.3 Future Scope

To build upon this work and address its current limitations, several avenues for future research and development are proposed:

- **Dynamic Surrogate Modeling:** Expanding the dataset to include transient data, allowing the surrogate to model dynamic process behaviors, startup/shutdown sequences, and advanced process control systems.
- **Active Learning Integration:** Implementing active learning pipelines where the model automatically queries the rigorous DWSIM simulator only when encountering high-uncertainty regions, optimizing training data efficiency.
- **Plant-Wide Hybrid Architecture:** Integrating the unit-level surrogate into an open-source plant-wide flowsheet to test real-time convergence and stability within complex recycle loops.

Reference

- DWSIM Project, FOSSEE, IIT Bombay: https://dwsim.fossee.in/
- PyTorch Documentation: https://pytorch.org/docs/
- scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
- Luca Mencarelli, Alexandre Pagot, Pascal Duchêne. Surrogate-based modeling techniques with application to catalytic reforming and isomerization processes. Computers & Chemical Engineering, 2020, 135, pp.106772. https://doi.org/10.1016/j.compchemeng.2020.106772
- Li, H.; Zhao, Q.; Wang, R.; Xu, W.; Qiu, T. Integrated Hybrid Modelling and Surrogate Model-Based Operation Optimization of Fluid Catalytic Cracking Process. Processes 2024, 12, 2474. https://doi.org/10.3390/pr12112474