Systems & Control Transactions

Research Article - Peer Reviewed Conference Proceeding
ESCAPE 35 - European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale, Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Surrogate Modeling of Twin-Screw Extruders Using a Recurrent Deep Embedding Network

Po-Hsun Huang[a], David Shan-Hill Wong[a], Yen-Ming Chen[b], Chih-Yu Chen[b], Meng-Hsin Chen[b], and Yuan Yao[a]*

[a] Department of Chemical Engineering, National Tsing Hua University, Hsinchu 30013, Taiwan
[b] Industrial Technology Research Institute Material and Chemical Research Laboratories, Hsinchu 30013, Taiwan
* Corresponding Author: yyao@mx.nthu.edu.tw.

ABSTRACT

Optimizing twin-screw extruder (TSE) performance is critical in the plastics industry but is often resource-intensive. This study introduces a novel surrogate modeling approach using a Recurrent Deep Embedding Network (RDEN) that integrates deep autoencoders with recurrent neural networks to capture sequential dependencies and physical relationships in TSE processes. Leveraging Progressive Latin Hypercube Sampling (PLHS), the RDEN achieves robust predictions of key process variable, like mean residence time. Results demonstrate the model's accuracy, generalization capabilities, and potential for automated screw design optimization.

Keywords: twin-screw extruder, deep learning, surrogate modeling

INTRODUCTION

Twin-screw extruders (TSEs) are extensively used in the plastics processing industry, with their performance highly dependent on operating conditions and screw configurations. However, optimizing these parameters through experimental trials is often time-consuming and resource-intensive. Although some neural network models have been proposed to tackle the screw arrangement problem [1], they didn't account for the physical and positional information of the screw elements. To overcome this challenge, we propose a recurrent deep embedding network (RDEN) that leverages a deep autoencoder with a recurrent neural network (RNN) structure to develop a surrogate model based on numerical simulation data [2, 3].

The details are as follows [4]. An autoencoder is a neural network architecture designed to learn latent representations of input data. In this study, we integrate the autoencoder with an RNN to capture the complex physical relationships between the operating conditions, screw configurations of TSEs, and their corresponding performance metrics. A schematic of TSEs simulation system is depicted in **Figure 1**. To further enhance the model's ability to represent screw positions, we incorporate an RNN layer and this addition allows the model to more effectively capture the spatial relationships between the screw elements.

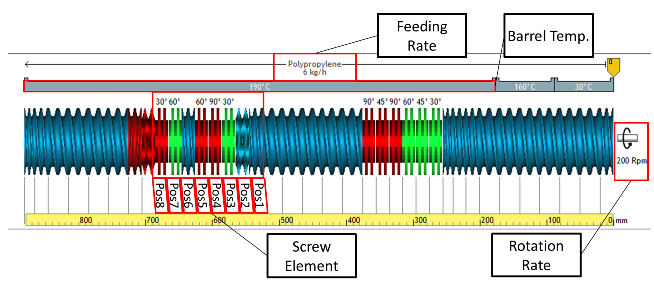

Figure 1. Schematic of TSEs simulation from Ludovic

METHODOLOGY

In this section, we would introduce the methods used for data collection and RDEN model building.

Progressive Latin Hypercube Sampling (PLHS)

Space filling is essential for sampling methods when working with small datasets, as it ensures optimal coverage of the parameter space while maintaining relatively few sample points. To address the problem, we apply Progressive Latin Hypercube Sampling (PLHS) [5].

PLHS is an extension of Latin Hypercube Sampling

(LHS), designed to enhance sampling efficiency in environmental modeling. Unlike LHS, which generates the entire sample set at once, PLHS progressively creates a series of smaller subsets while striving to maintain LHS characteristics in each subset. This approach is particularly advantageous for large and complex environmental models, offering improved stratification and more accurate representation of the input space.

To understand PLHS, we first describe the LHS method. For LHS, let $S(n,p)$ be a sample matrix containing $(n \times p)$ elements $x_{i,j} \in [0,1]$, where $i = 1,\dots,n$ and $j = 1,\dots,p$. The factor space $[0,1]^p$ is divided into n non-overlapping intervals $([0,1/n),[1/n,2/n),\dots,[(n-1)/n,1])$, with each interval indexed by q along each dimension, where $q = 1,\dots,n$. To represent the relationship between sampling points and intervals, we define an auxiliary binary variable $y_{q,j}$ as follows:

$$y_{q,j} = \begin{cases} 1 & \begin{gathered} \textit{if there exists any } i \\ \textit{such that } x_{i,j} \textit{ lies within interval } q \end{gathered} \\ 0 & \textit{otherwise} \end{cases} \tag{1}$$

The sample matrix $S(n,p)$ is considered a Latin hypercube when the following condition is met:

$$\frac{1}{n \cdot p} \sum_{j=1}^{p} \sum_{q=1}^{n} y_{i,j} = 1 \tag{2}$$

This condition ensures that each interval is sampled exactly once in each dimension, maintaining the Latin hypercube property.

For PLHS, we extend this formulation by defining an auxiliary binary variable $y_{q,j}^t$ corresponding to the sample matrix $L^t = \bigcup_{k=1}^{t} S_k(n_k,p)$, where $t = 1,2,\dots,T$ represents the slice number. The number of sample points is $n_t = \sum_{k=1}^{t} n_k$, and the factor space $[0,1]^p$ is divided into n_t non-overlapping intervals. The necessary and sufficient condition for the sample to be a progressive Latin hypercube is:

$$\frac{1}{n_t \cdot p} \sum_{t=1}^{T} \sum_{j=1}^{p} \sum_{q=1}^{n_t} y_{q,j^t} = T \tag{3}$$

The left side of the equation represents the sum of Latin hypercube properties across all T slices. Thus, generating a PLHS can be expressed as an optimization problem:

$$\max \left(\sum_{t=1}^{T} \frac{1}{n_t \cdot p} \sum_{j=1}^{p} \sum_{q=1}^{n_t} y_{q,j}^t \right) \tag{4}$$

subject to the Latin hypercube constraints:

$$\begin{aligned} &\sum_{i=1}^{n_t} x_{i,j} = 1 \qquad \forall t, \forall j \\ &y_{q,j}^t \in 0,1 \qquad \forall t, \forall q, \forall j \end{aligned} \tag{5}$$

where $x_{i,j}$ are decision variables representing sample point regions, and $y_{q,j}^t$ are auxiliary variables ensuring the Latin hypercube property.

The PLHS method offers several advantages for data collection. It allows for incremental addition of sample points, adapting to different computational budgets and analysis needs. By generating samples progressively, PLHS can better cover the input space with fewer points, avoiding repeated sampling in the same regions and reducing computational costs, and the workflow is shown in **Figure 2**.

In the workflow, PLHS enables progressive construction of RDEN dataset while allowing for additional constraints, ensuring desired stratification properties in each subset and combined set in further practical process applications.

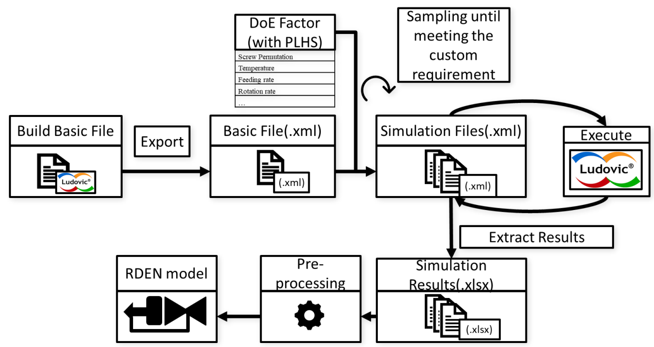

Figure 2. Workflow of dataset and model

DEN with RNN Structure

The DEN has emerged as a powerful technique in TSE modeling, demonstrating the ability to extract latent physical information. However, the original model faces challenges of capturing sequential dependencies which are inherent in extruder element arrangements.

To address this limitation, we propose integrating the DEN with an RNN structure. Specifically, the input to the autoencoder structure consists of a series of screw elements, denoted as z, with each element represented in one-hot encoded format. The autoencoder reduces the dimensionality of the coding of each screw element while reconstructing it as $\hat{z}$. To ensure the latent variables extracted from different screw positions remain physically consistent, the autoencoder is designed to share the same parameters across all screw inputs.

For the prediction task, we employ Gated Recurrent Unit (GRU) cells to process the sequential arrangement of extruder elements and their associated process conditions. The GRU utilizes the latent variables from the autoencoder, denoted as h, alongside the process conditions, represented by c. This integrated input allows the model to effectively capture sequential trends, enabling the transfer of information from earlier extruder elements to subsequent GRU cells.

The combined RDEN structure offers enhanced flexibility and scalability, making it well-suited for various extrusion process configurations. The model structure and pseudo code are depicted in **Figure 3** and the following section separately.

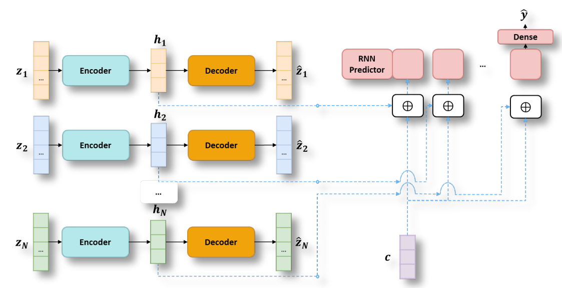

Figure 3. RDEN model structure

```
START ALGORITHM
FOR iter = 1 TO max_training_iter DO
    batch ← SAMPLE_BATCH(training_data)
    FOR EACH ((z1, …, zN), c, y) IN batch DO
        // AutoEncoder and reconstruct
        FOR j = 1 TO Pos DO
            h_j ← ENCODER(z_j)
            z_j_recon ← DECODER(h_j)
            L_recon ← L_recon + |z_j, z_j_recon|²
        END FOR
        // Predict output
        y_pred ← PREDICTOR(h1, …, hN, c)
        L_pred ← L_pred +|y, y_pred|²
    END FOR
    // Update parameters
    L_total ← L_recon + L_pred
    UPDATE_PARAMETERS(ENCODER, DECODER, PREDICTOR)
END FOR
END ALGORITHM
```

The model outputs include predictions of materials' final physical properties, capturing the dynamic nature of the extrusion process, as well as the autoencoder reconstruction. For data normalization, the dataset is reshaped into three dimensions: samples, position series, and process variables; then, we apply the min-max scaling to inputs and the z-score to output, along the variables' direction. Such normalization ensures that the model preserves output trends without necessitating extensive hyperparameter tuning. The model's loss functions, incorporating both reconstruction and prediction errors, are mathematically represented in (6) and calculated using the mean squared error.

$$\begin{gathered} L_1 = E\left(\sum_{i=1}^{n}(z_i - \hat{z}_i)^T(z_i - \hat{z}_i)\right) \\ L_2 = E((y_i - \hat{y}_i)^T(y_i - \hat{y}_i)) \end{gathered} \tag{6}$$

RESULTS AND DISCUSSIONS

We successfully established the settings for the Ludovic simulation system and generated data based on the experimental design (DoE) table, presented in **Table 1**. The Ludovic system simulates a twin-screw extruder that models material flow, temperature, pressure, and residence time across the screw configuration. By incorporating detailed screw configurations and process conditions, the Ludovic simulation provides a detailed representation of the twin-screw extruder (TSE) process, modeling material flow, temperature distribution, pressure variations, and residence time across different screw configurations.

Table 1: DoE Table

Factor		Unit	Design Range
Screw Element	Position: [525, 545]	-	"20-20-L2", "20-20-R2", "KB-20-5-30a-L2", "KB-20-5-30a-R2", "KB-20-5-45a-L2", "KB-20-5-45a-R2", "KB-20-5-60a-L2", "KB-20-5-60a-R2", "KB-20-5-90a-L2", "KB-20-5-90a-R2", "SME-20-20-R2", "ZME-20-20-L2"
	Position: [545, 565]	-	
	Position: [565, 585]	-	
	Position: [585, 605]	-	
	Position: [605, 625]	-	
	Position: [625, 645]	-	
	Position: [645, 665]	-	
	Position: [665, 685]	-	
Temp	Position: [525, 685]	°C	[180, 220]
Rotation rate	Position: [525, 685]	rpm	[200, 300]
Feeding rate	Position: [525, 685]	kg/hr	[15, 25]

This study focuses on a system with eight adjustable screw element positions, as illustrated in Figure 1, where key design factors—the 8-position screw arrangement, feeding rate, barrel temperature, and rotation rate—are highlighted with red boxes. The arrangement is varied using a set of candidate screw elements, shown in **Figure 4**, including conveying elements (20-20-L2, 20-20-R2), kneading blocks with various staggering angles (KB-20-5-30a, KB-20-5-45a, KB-20-5-60a, KB-20-5-90a), and specialized mixing elements (SME-20-20-R2, ZME-20-20-L2). Building on this foundation, we systematically explored various element arrangements and analyzed their impacts on the process variables.

Detailed investigations into mean residence time were conducted, offering insights into their interdependencies and contributions to overall process efficiency. Furthermore, the incorporation of progressive Latin hypercube sampling provided a robust framework to optimize the sampling process, enabling comprehensive exploration of the factor space with proper sampling size.

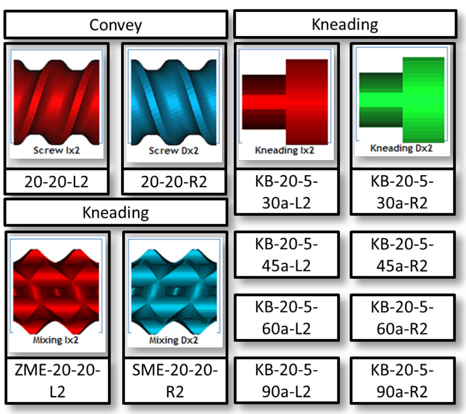

Figure 4. Schematic of Ludovic Screw Element

The objective of this study is to predict a critical process variable, the mean residence time, across the entire screw configuration system. Specifically, the model is designed to analyze an eight-position screw assembly, with each position containing a number of candidate screw elements.

To ensure the robustness and generalization of our model, the dataset was meticulously divided into training, validation, and testing sets, adhering to proportions as detailed in **Table 2**.

Table 2. Dataset Deviation

	Sample Size	Percentage
Training set	14386	0.85
Validation set	1690	0.10
Test set	844	0.05

For model training, we employ the RDEN framework described in the previous section to process the training set, guided with a carefully designed loss function. The parameters of both the autoencoder and the predictor are trained simultaneously on the training data. Additionally, the validation set is employed to fine-tune the model and mitigate the risk of overfitting.

The training process is managed using several callback functions to enhance model performance and ensure proper training. Specifically, we use early stopping to halt training when the model's performance on the validation set stops improving. Model checkpoints are used to save the model weights at each epoch, ensuring that the best performing model is retained. Furthermore, the best performing epoch is identified and recorded, helping to capture the model's peak performance.

The results demonstrate that the model performs exceptionally well across all three datasets, with R^2 scores over 0.99 and RMSE values below 1, which is within the commonly accepted tolerance for mean residence time. These findings highlight the model's accuracy. The final prediction results are presented below.

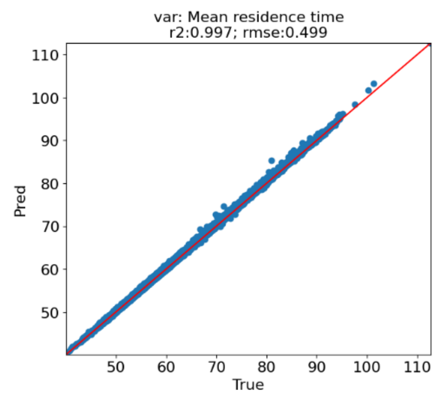

Figure 5. Parity plot of the training dataset for mean residence time

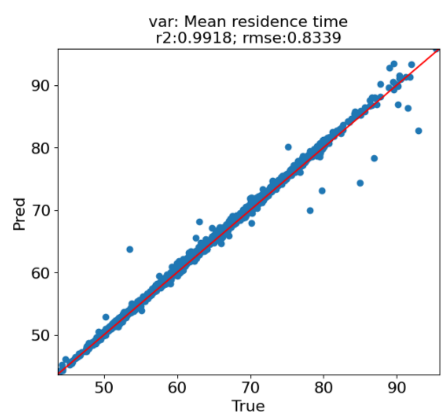

Figure 6. Parity plot of the validation dataset for mean residence time

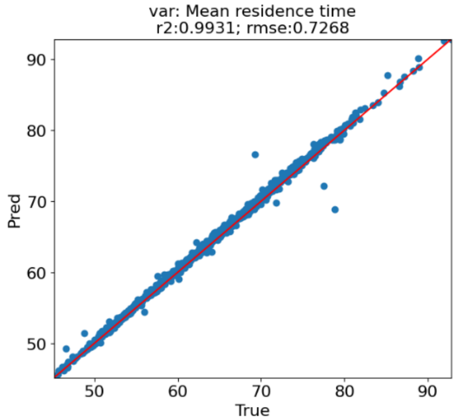

Figure 7. Parity plot of the test dataset for mean residence time

In TSE systems, screw elements are typically categorized according to their functionality. Conveying elements, such as 20-20-L2 and 20-20-R2, are primarily responsible for material transportation. Kneading elements, including KB-20-5-30a-L2, KB-20-5-45a-L2, KB-20-5-60a-L2, KB-20-5-90a-L2, KB-20-5-30a-R2, KB-20-5-45a-R2, KB-20-5-60a-R2, and KB-20-5-90a-R2, are designed to apply shearing forces to the material for better dispersion and distribution. Mixing elements, represented by SME-20-20-R2 and ZME-20-20-L2, enhance material blending through intensive mechanical mixing.

The latent space visualization in **Figure 8**. demonstrates distinct clustering patterns, revealing the model's ability to capture inherent physical characteristics of screw elements. For instance, a clear spatial separation was observed between left- and right-rotating elements. Additionally, the demarcation between kneading and conveying elements indicates that the learned representations align with the mechanical principles governing the extrusion process. These systematic clustering patterns across different screw configurations validate the model's capability to encode physical phenomena and provide insights into the functional relationships between various screw elements in the system. By analyzing these patterns, we gain a deeper understanding of the interrelations between the variables, emphasizing the model's capability to elucidate the intricate mechanisms governing the process.

CONCLUSIONS

In this paper, we introduced a novel model, RDEN, which synergizes the strengths of Deep Embedding Networks and Recurrent Neural Networks. This hybrid architecture exhibits two key capabilities. First is physical information Extraction. The DEN component effectively captures latent physical information from the extrusion process data. Second is sequential prediction. The RNN structure accurately predicts sequential variables inherent in the arrangement of extruder elements. This is achieved through a three-dimensional data structure that significantly reduces redundancy in the dataset and training process, thereby enhancing computational efficiency without compromising performance.

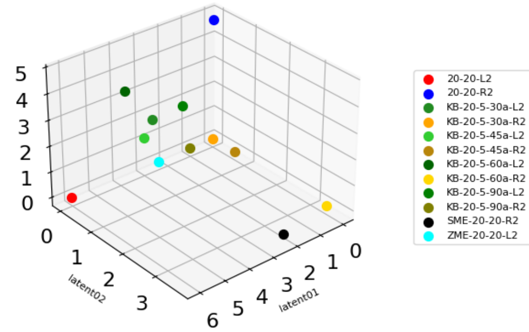

Figure 8. Scatter plot of latent variables

The predictive results for process variables demonstrate the model's reliability as a surrogate for extruder systems. Additionally, the latent screw representations learned by the model reveal meaningful patterns in the extrusion process, highlighting its potential for generating valuable process insights.

While our current work establishes RDEN as an effective modeling approach, the development of an optimization framework that leverages this model to recommend optimal screw arrangements remains a promising direction for future research, rather than a validated finding of this study. Our ongoing efforts will focus on exploring this potential application to further establish RDEN as a versatile and powerful tool for advanced process modeling and optimization in polymer extrusion and related fields.

REFERENCES

1. Kowalski, R.J., E. Pietrysiak, and G.M. Ganjyal, Optimizing screw profiles for twin-screw food extrusion processing through genetic algorithms and neural networks. *J. Food. Eng.*, 303. (2021) https://doi.org/10.1016/j.jfoodeng.2021.110589
2. Yi L, Glatt M, Ehmsen S, Duan W, Aurich JC., Process monitoring of economic and environmental performance of a material extrusion printer using an augmented reality-based digital twin. *Addit. Manuf.*, 48. (2021) https://doi.org/10.1016/j.addma.2021.102388
3. Corradini, F. and M. Silvestri, Design and testing of a digital twin for monitoring and quality assessment of material extrusion process. *Addit. Manuf.*, 51. (2022)

https://doi.org/10.1016/j.addma.2022.102633
4. Wu H, Lo YH, Zhou L, Yao Y., Process modeling by integrating quantitative and qualitative information using a deep embedding network and its application to an extrusion process. *J. Process Control,* 115: p. 48-57. (2022) https://doi.org/10.1016/j.jprocont.2022.04.018
5. Sheikholeslami, R. and S. Razavi, Progressive Latin Hypercube Sampling: An efficient approach for robust sampling-based analysis of environmental models. *Environ. Model Softw.*, 93: p. 109-126. (2017) https://doi.org/10.1016/j.envsoft.2017.03.010