mathematics

MDPI

Article

A Novel Methodology for Performance Evaluation in Advanced Quality Control

Ethel García, Rita Peñabaena-Niebles *,†, Winston S. Percybrooks † and Kevin Palomino †

Department of Industrial Engineering, Universidad del Norte, Barranquilla 081007, Colombia; ethelg@uninorte.edu.co (E.G.); wpercyb@uninorte.edu.co (W.S.P.); krpalomino@uninorte.edu.co (K.P.)

* Correspondence: rpena@uninorte.edu.co

† These authors contributed equally to this work.

Abstract: Current global conditions and challenges in industrial manufacturing, marked by dynamism, competition, and the need for responsible resource management, have increased the demand for sustainable manufacturing practices. The integration of Industry 4.0 and the recent development of Industry 5.0 have added dynamism, which has generated profound implications for quality control and process monitoring, focusing mainly on recognising control patterns within the manufacturing environment. This study introduces a novel methodology for evaluating the performance of pattern classification models used in advanced quality control. Our approach incorporates robust performance metrics, early detection, window size, network hyperparameters, and concurrent patterns within a simulated monitoring environment. Unlike previous research, our evaluation methodology addresses the sensitivity of classification models to various factors, emphasising the critical balance between early detection and minimising false alarms. The findings reveal that window size significantly impacts the model's sensitivity to pattern changes, highlighting that measuring early detection alone is impractical in real-world applications. Furthermore, optimal hyperparameter selection enhances the model's practical applicability.

Keywords: control chart pattern recognition; concurrent pattern; window size; long short term memory

MSC: 62P30; 68T05; 37M10



Academic Editors: Qing Li, Bing Si and Renjie Hu

Received: 25 November 2024
Revised: 17 December 2024
Accepted: 19 December 2024
Published: 14 January 2025

Citation: García, E.; Peñabaena-Niebles, R.; Percybrooks, W.S.; Palomino, K. A Novel Methodology for Performance Evaluation in Advanced Quality Control. *Mathematics* **2025**, *13*, 259. https://doi.org/10.3390/math13020259

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Current global conditions and challenges in industrial manufacturing, marked by dynamism, competition, and the need for responsible resource management, have increased the demand for sustainable manufacturing practices [1]. The integration of Industry 4.0 and the recent development of Industry 5.0 have added to this dynamism, which has generated profound implications for quality control and process monitoring, focusing mainly on control pattern recognition within the manufacturing environment.

Control charts have traditionally been the primary tool in statistical process control (SPC), consisting of two control limits and a sequence of observations plotted during the monitoring process. However, the effectiveness of these charts for pattern identification is limited, which has led to the development of rules of thumb that, although practical, require expert knowledge and do not accurately discriminate the target pattern in the process observations [2,3].

The main advantage of pattern recognition in control charts lies in the diagnostic procedure; an abnormal pattern may be associated with a specific assignable cause, reducing

the search time [4,5]. This recognition signals the existence of a fault and facilitates the fault elimination process [6,7]. Some patterns associated with failures in production processes are documented in studies such as those of [2,8–11].

Industry 4.0 and 5.0, driven by advanced technologies such as the Internet of Things (IoT), big data, and artificial intelligence (AI), hold significant potential to revolutionise quality monitoring through real-time data acquisition and analysis. However, the effectiveness of these innovations relies heavily on developing robust and accurate pattern recognition algorithms. Over the past two decades, machine learning (ML)-based approaches for control chart pattern recognition (CCPR) have gained substantial attention and have been extensively applied [12,13]. Examples include models based on artificial neural networks (ANNs) [13–18], decision trees (DTs) [19,20], support vector machines (SVMs) [21–25], unsupervised learning frameworks [26–30], hybrid schemes combining multiple machine learning models [31–33], and others [10,34–37].

Despite significant advances in applying machine learning technologies for pattern recognition in quality monitoring, there are significant gaps in the existing literature, especially in sustainable manufacturing under the Industry 4.0 and 5.0 paradigms. One of these challenges relates to the simulation of training and testing datasets, a standard process in pattern recognition due to the complexity of data collection in actual processes [38]. Refs. [11,38–40] have pointed out that most studies on pattern recognition simulate complete data windows (see Figure 1), which can affect early detection capabilities. As a solution, Ref. [5] proposes a technique for generating synthetic data by simulating the gradual introduction of unnatural patterns.

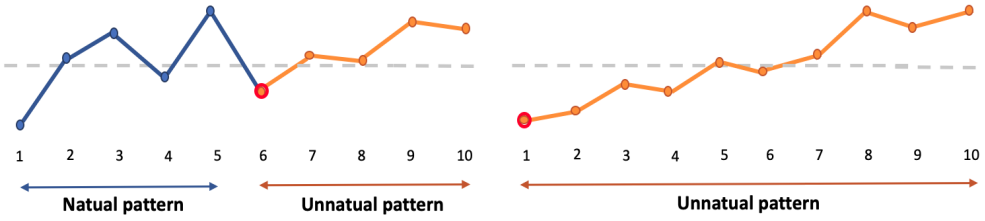


Figure 1. The gradually displayed pattern (**left**) vs. the fully displayed pattern (**right**).

However, alongside this simulation problem, there is a pressing need to improve the performance evaluation process of monitoring models. Ref. [40] noted that most research on concurrent pattern recognition relies solely on precision metrics, even though the primary objective of these models is to detect anomalous patterns as early as possible while minimising false alarms. Consequently, it is common in the literature to find detection models with high precision rates that may not be effective in real-world environments [5].

In traditional fault detection, run length (ARL) and time to signal (ATS) are the two most important performance measures of control charts. However, these metrics are rarely used in concurrent pattern recognition. Ref. [41] proposed adaptations of ARL for concurrent pattern recognition. Similarly, Ref. [39] introduced the Unified Score metric to consolidate multiple performance measures.

In contrast to existing evaluation approaches, this research proposes estimating model performance by analysing the trade-off between early fault detection and false alarm minimisation. In real processes, late detections generate costs associated with operating under suboptimal conditions, while false alarms lead to unnecessary production stoppages. Balancing these two metrics is essential for an effective monitoring process. However, current studies often incorporate false alarms as part of a global metric linked to pattern detection accuracy—whether natural or unnatural—without recognising false alarms as an equally critical metric compared to early detection.

In our approach, we propose evaluating false alarms and early detection as two independent variables that must be balanced. The significance of this new evaluation framework lies in the long periods without anomalies during the monitoring process. Consequently, the accuracy of detecting natural patterns becomes increasingly important. Giving equal weight to the detection accuracy of both natural and unnatural patterns may result in an elevated number of false alarms due to the higher prevalence of the in-control state. Furthermore, if care is not taken, prioritising early detection will lead to an increased number of false alarms in the rush to quickly identify changes, thus amplifying errors during the natural state of the process. The proposed approach aims to address this shortcoming by jointly balancing these two characteristics.

In this way, seven evaluation metrics are proposed to estimate monitoring models' accuracy, early detection, and false alarms. In addition, the proposed model evaluation methodology seeks to determine the model's sensitivity to variations in key hyperparameters, such as window size and hidden layer size, and to identify the conditions that best maintain a balance between early fault detection and false alarm minimisation. It also seeks to determine the model's ability to detect each specific pattern to establish the pattern's effect on classification performance individually. Identifying which patterns are more complex to identify may be desirable in real applications. Finally, the complexity of the model architecture may be a factor that differentiates different classification proposals. In the present proposal, complexity will be evaluated through training time and with the ratio of parameters versus training sample size.

Beyond identifying an optimal performance value, the proposed methodology presents multiple performance scenarios under different window and hidden layer sizes. Thus, the optimal operating conditions will inherently depend on the specific process being monitored and the established priorities. In this regard, a specific model configuration could be selected if the goal is to detect failures rapidly while accepting a certain level of false alarms. This methodology is particularly relevant within the context of Industry 5.0, as it enables the customisation of the model to the unique characteristics of a specific process, driving the reduction in operational costs associated with waste and rework while pursuing more efficient resource utilisation to support sustainability efforts.

This evaluation methodology is intended to measure model performance in a comprehensive way (see Figure 2), as well as to facilitate the comparative process between different pattern monitoring proposals. To present the new model evaluation methodology, a deep neural network (DNN) model will be designed to recognise concurrent control patterns. In the context of Industry 4.0 and 5.0, developing DNN-based monitoring models is relevant to managing automated processes. These processes are characterised by data collection through sensors, which increases complexity due to the large volume of data generated in real time. In contrast to traditional models that dominate most current research and fail to extract relevant data features automatically, DNNs offer a promising solution to overcome these limitations. In this study, we will use a specialised variant of DNNs known as a long short-term memory (LSTM) network, which is specifically designed to address the long-term dependencies present in sequential data. Although these schemes are effective for control pattern recognition, few studies address concurrency in LSTM patterns and models (Table 1).

This document is structured as follows to explore the application of deep learning in quality control pattern recognition, which is crucial for Industry 4.0 and 5.0. Section 2 outlines the methodology used for the evaluation process, which is tailored to the challenges of modern manufacturing environments. Section 3 describes generating simple and concurrent patterns, reflecting production complexities. Section 4 details the training and testing procedure of the model, preparing it for real-world variations. The results

presented in Section 5 validate the approach's effectiveness and suggest areas for process optimisation. Section 6 discusses the challenges in pattern identification, indicating the need for evolution in monitoring techniques. Finally, Section 7 summarises the study's key findings, underscoring its contribution to sustainable and efficient manufacturing practices in alignment with the principles of Industry 4.0 and 5.0.

Table 1. Summary of LSTM schemes in the literature.

Study	Proposed Scheme	Pattern Type	Data Type	Input Data	Online	Metrics
[42]	Bidirectional LSTM (Bi-LSTM)	Simple	Independent and identically distributed (IID)	Raw data	No	Correct recognition rate (CRR)
[43]	LSTM and multivariate adaptive regression splines (MARS)	Concurrent	Autocorrelated	Data after engineering process control (EPC)	No	Accuracy identification rate (AIR)
[11]	Cost-oriented LSTM	Simple	IID	Raw data	Yes	Accuracy, sensitivity, specificity, G-mean and early detection metrics
[44]	Convolutional auto-encoder (CAE) and CNN-LSTM decoder	Concurrent	IID	Raw data, statistical and shape features	No	Accuracy
[39]	Bidirectional LSTM	Simple	IID	Raw data	Yes	Unified Score

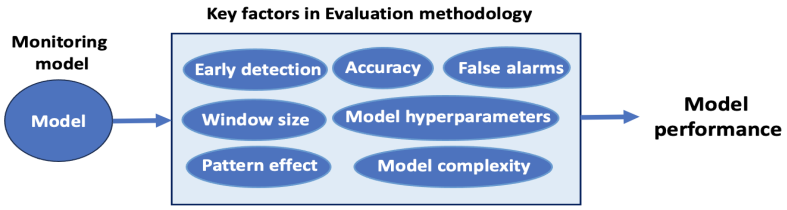


Figure 2. Methodology for performance evaluation in advanced quality control models.

2. Methodology

As we delve into the methodological section of our study, we present a multidimensional approach that spans conceptual reviews of recurrent neural networks—particularly LSTMs—including network architecture design, window size selection, and the definition of evaluation metrics. This methodological framework enables us to thoroughly explore our analysis's technical intricacies, examining the relationship between model configuration and its performance in detecting patterns within advanced manufacturing environments. The following subsections provide a detailed discussion of key technical aspects, from the internal structure of LSTM cells to the proposed evaluation procedures, each contributing to a comprehensive framework for optimising quality monitoring systems in the context of Industry 4.0 and 5.0.

2.1. Overview of LSTMs

A Recurrent Neural Network (RNN) is a specialised architecture of artificial neural networks (ANNs) designed to process sequential information. However, conventional RNNs often need help with learning issues, such as exploding and vanishing gradients. In response to these challenges, researchers have developed improved versions of RNNs, including the LSTM network. This network introduces a novel memory state designed

to mitigate gradient problems, enabling it to learn complex patterns like seasonality, autocorrelation, trends, and noise in time-series data. Consequently, LSTM networks are extensively used in time-series prediction problems.

Figure 3 presents an LSTM cell, where f_t and i_t represent forget and input gates, respectively. These gates decide which information to forget and which to add to the cell state C_t.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{1}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2}$$

where x_t represents the present input data; f_t and i_t denote the output from the forget gate and input gate, respectively; the U_f and W_i variables are weight matrices; b is the bias vector, also called the thresholds of the input gate b_i and the thresholds of the forget gate b_f; and $\sigma(.)$ represents the sigmoid function defined as:

$$\sigma(z) = \frac{1}{1 + exp(-z)} \tag{3}$$

The $\sigma(.)$ output is in the range of 0 to 1. The $\widetilde{C}_t$ value is defined as:

$$\widetilde{C}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{4}$$

where $\widetilde{C}_t$ is the new candidate cell state, called the internal state of the cell, and b_c is the threshold of the memory cell.

Then, the cell state C_t vector is updated as follows:

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \widetilde{C}_t \tag{5}$$

The O_t value is the output obtained as:

$$O_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{6}$$

where b_o represents the threshold of the output gate. Finally, the hidden state, h_t, of the LSTM can be derived as:

$$h_t = O_t \otimes tanh(C_t) \tag{7}$$

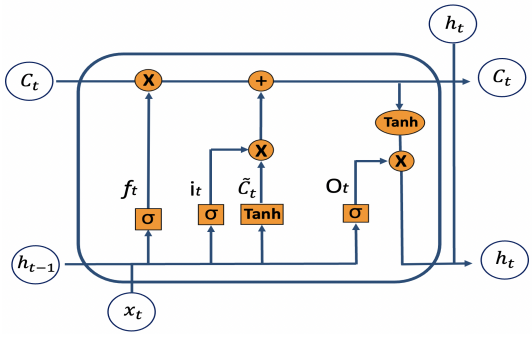


Figure 3. The hidden layer structure of LSTM.

Figure 4 illustrates the LSTM forward pass for a single analysis window, displaying N LSTM cells, one for each observation, and how the output of each cell feeds into the next. In an LSTM network, the hidden size parameter corresponds to the dimension of the vector h_t, and the number of layers represents how many cells are vertically arranged, with a single

layer size depicted in the Figure 4 . The LSTM learning process involves computing a loss function followed by backpropagation, where errors are disseminated using gradients.

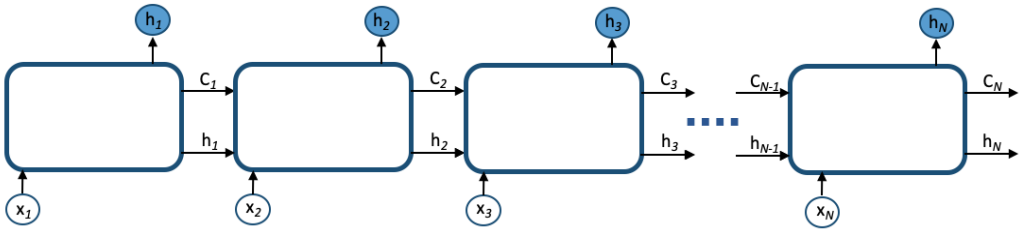


Figure 4. The LSTM network for a single window.

2.2. Network Architecture

As mentioned, the LSTM network will be tested with various windows and hidden sizes, keeping other parameters constant during each experiment. Built using the PyTorch library and Google Collaborator as the development environment, the model was trained in one of Google Colaboratory's GPU environments.

The network architecture, depicted in Figure 5, shows how input data feed the LSTM cells and, after the forward pass, the fully connected layer (FC) receives the last LSTM output (h_N) for prediction. The FC's output size is set to 16, with accuracy as the sole training metric. Optimization was achieved using an Adam optimiser and the cross-entropy loss function. The learning rate was set at 0.001, with early stopping based on a validation set of 896,400 samples. The batch size and the number of layers were set at 11,205 and 1, respectively.

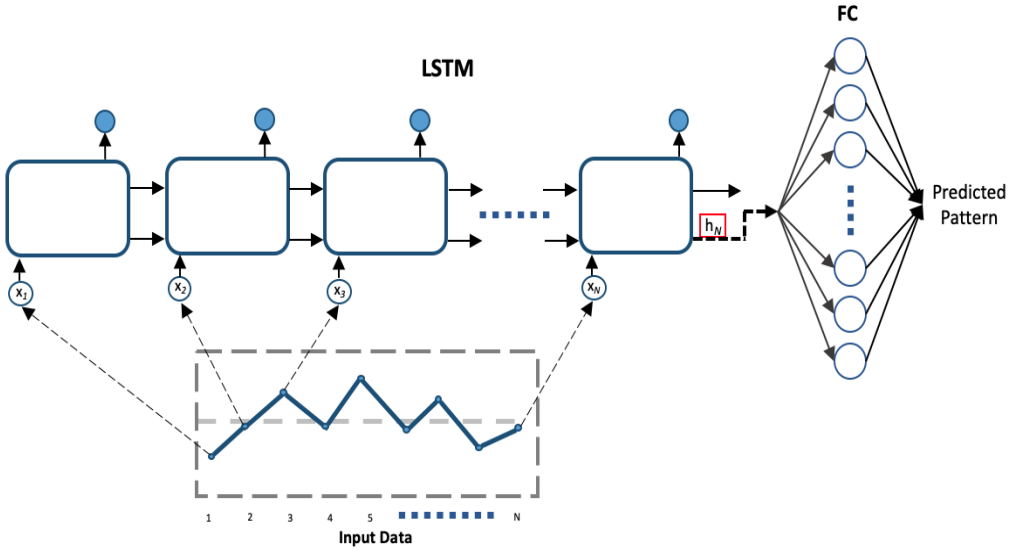


Figure 5. The LSTM network architecture.

2.3. Window Size and Evaluation Metrics

In a traditional monitoring strategy, samples are taken at each time step t and plotted on a control chart. This process continues until an unnatural pattern is detected, leading to the identification and elimination of its root cause before restarting the monitoring. Traditionally, the state of the process is assessed by evaluating the last N samples, referred to as the window size. As a new sample arrives, the oldest is removed, a process depicted in Figure 6 as a moving window, necessitating the evaluation of a pattern recognition model to select an optimal window size for classification metrics.

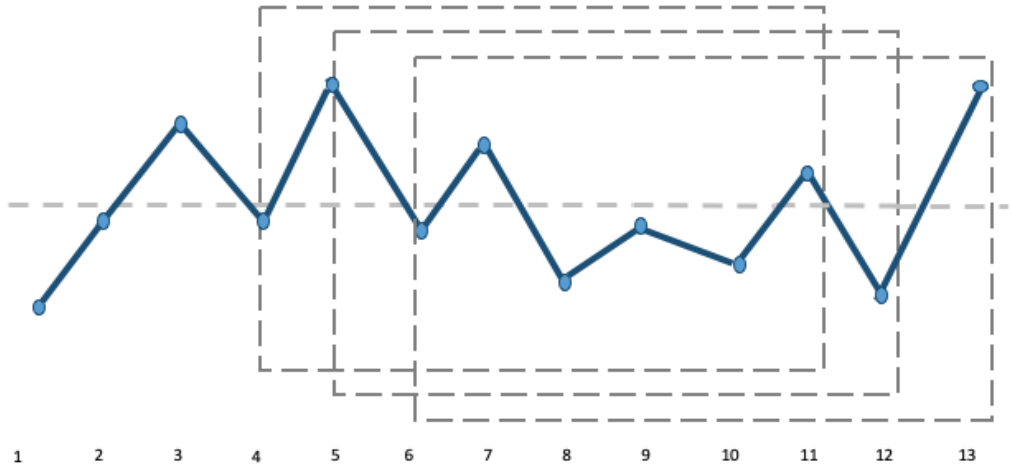


Figure 6. Illustration of moving window.

Furthermore, evaluation metrics play a crucial role in control chart recognition schemes. The study by [41] highlighted that single pattern recognition metrics are insufficient for concurrent pattern recognition schemes, proposing additional metrics for testing despite most studies focusing solely on accuracy and sometimes on Average Run Length (ARL). This study will include metrics associated with early detection, accuracy, and false alarms. To evaluate early detection, the following metrics will be utilised:

1. Average Target Pattern Run Length (ATPRL): The average number of samples needed to detect a target pattern.
2. Average Unnatural Pattern Run Length (AUPRL): The average number of samples required to detect an unnatural pattern, regardless of whether it is the target.

The ATPRL will help determine how quickly the target pattern is detected once a fault occurs. However, it is also important to assess the model's ability to detect any pattern change, whether or not it is the target pattern, as this indicates that something is affecting the natural operation of the process. This approach could help avoid production under suboptimal conditions, even if it does not reduce the search time associated with correctly identifying the fault.

On the other hand, the Detection Average (DA) metric will be used to analyse the model's accuracy:

1. Detection Average (DA): The average number of moving windows where the unnatural target pattern was detected.

This metric will estimate the model's ability to detect the presence of unnatural patterns in the moving windows. Finally, four metrics are proposed to estimate false alarms:

1. Average False Alarm (AFA): The average number of false alarms per moving window.
2. Number of false alarm moving windows per total moving windows (NFAMW-TMW): The total number of moving windows with at least one false alarm (NFAMW) divided by the total number of moving windows (TMW).
3. Number of false alarms per total moving windows (NFA-TMW): The total number of false alarms (NFA) divided by the total number of moving windows (TMW).
4. Number of false alarms per number of false alarm moving windows (NFA-NFAMW): The total number of false alarms (NFA) divided by the number of moving windows with at least one false alarm (NFAMW).

The AFA metric will estimate false alarms, which correspond to incorrect signals when no faults are present. Meanwhile, NFA-NFAMW, NFA-TMW, and NFAMW-TMW are ratios that establish the number of false alarms per moving window.

Pattern detection performance is also relevant in recognition models. The primary objective of pattern identification is to classify the pattern present in the monitoring data and associate it with a specific assignable cause. While identifying the exact pattern is a priority, we acknowledge that models are imperfect, and narrowing the search space can also benefit real-world environments. For this reason, it is proposed that patterns that are confused with one another be evaluated using a confusion matrix. This reduces the search process to one or two fault options. In this regard, stable models with confusion limited to fewer patterns are preferred. Determining the ATPRL for each pattern will help identify which patterns are easier or more difficult for the monitoring model to detect.

2.4. Proposed LSTM Evaluation Procedure

The proposed methodology aims to assess how different window sizes, hidden network sizes (HN sizes), and unnatural pattern characteristics impact the performance of a model based on an LSTM network. Despite LSTM's ability to infer from variable-length input sequences, this methodology will focus on training and testing with fixed window lengths. Various models, each with its window size, will be considered for sensitivity analysis, with window sizes set at 8, 16, 24, 32, 40, and 48 samples.

In addition to window length, experiments will explore the impact of the LSTM's HN size on network performance, analysing how it affects classification performance and training time. HN size values for sensitivity analysis will be 30, 60, 120, 180, and 240.

Finally, LSTM's classification performance on simple and concurrent patterns will be evaluated, considering patterns such as Trend (TP), Shift (SP), Cycle (CP), Stratified (STP), and Systematic (SYS). Concurrent patterns will result from all possible combinations of simple patterns, as detailed in the following section.

3. Pattern Generation

The sensitivity analysis will include the simple patterns described in Table 2, detailing each pattern's name, estimation formula, and parameter range. The natural pattern will be generated based on a normally distributed process. Table 2, $p(t)$ represents the pattern observation at time t, while $r_i(t)$ denotes white Gaussian noise.

Table 2. Pattern generation.

Pattern	Parameter	Generator Equation	Parameter Values
Natural (NP)	-	$p_i(t) = \mu + r_i(t)\sigma$	$\mu = 0, \sigma = 1$
Trend (TP)	Slope (s)	$p_i(t) = \mu + r_i(t)\sigma + ts$	$s = (0.075\sigma, 0.1\sigma, 0.125\sigma)$
Shift (SP)	Shift magnitude (η)	$p_i(t) = \mu + r_i(t)\sigma + t\eta$	$\eta = (1.5\sigma, 2\sigma, 2.5\sigma)$
Cycle (CP)	Amplitude (α), Period (T)	$p_i(t) = \mu + r_i(t)\sigma + \alpha sin(2\pi/T)$	$\alpha = (1.5\sigma, 2\sigma, 2.5\sigma)$, T = (8,16)
Stratified (STP)	Random noise (σ')	$p_i(t) = \mu + r_i(t)\sigma'$	$\sigma' = (0.3\sigma, 0.5\sigma, 0.7\sigma)$
Systematic (SYS)	Systematic departure (d)	$p_i(t) = \mu + r_i(t)\sigma + d(-1)^t$	d = $(1\sigma, 2\sigma, 3\sigma)$

Concurrent patterns will be simulated by combining pairs of simple patterns and their respective parameter ranges, resulting in 16 patterns to analyse. This includes one natural pattern and fifteen unnatural patterns (5 simple and all 10 possible pairs). The variation in parameter numbers for each pattern may lead to an imbalanced dataset, which will be corrected as necessary through oversampling. Table 3 summarises the outcomes of the pattern generation process.

Table 3. Training and testing data.

Pattern Name	Parameter Before Balancing	Oversampling Factor	Parameter After Balancing	Windows			
				Training		Testing	
				Natural	Unnatural	Natural	Unnatural
1. Stratified (STP)	3	12	36	103,500	270,000	41,400	108,000
2. Cycle (CP)	6	6	36	103,500	270,000	41,400	108,000
3. Trend (TP)	6	6	36	103,500	270,000	41,400	108,000
4. Systematic (SYP)	3	12	36	103,500	270,000	41,400	108,000
5. Shift (SP)	6	6	36	103,500	270,000	41,400	108,000
6. Stratified-Cycle (STP-CP)	18	2	36	103,500	270,000	41,400	108,000
7. Stratified-Trend (STP-TP)	18	2	36	103,500	270,000	41,400	108,000
8. Stratified-Systematic (STP-SYP)	9	4	36	103,500	270,000	41,400	108,000
9. Stratified-Shift (STP-SP)	18	2	36	103,500	270,000	41,400	108,000
10. Cycle-Trend (CP-TP)	36	1	36	103,500	270,000	41,400	108,000
11. Cycle-Systematic (CP-SYP)	18	2	36	103,500	270,000	41,400	108,000
12. Cycle-Shift (CP-SP)	36	1	36	103,500	270,000	41,400	108,000
13. Trend-Systematic (TP-SYP)	18	2	36	103,500	270,000	41,400	108,000
14. Trend-Shift (TP-SP)	36	1	36	103,500	270,000	41,400	108,000
15. Systematic-Shift (SYP-SP)	18	2	36	103,500	270,000	41,400	108,000
	TOTAL			1,552,500	4,050,000	621,000	1,620,000

4. Simulation Approach for Training and Testing Data

The proposed simulation method is similar to [5]. However, unlike those authors, the analysis windows will be generated by removing the oldest sample and including a new sample. In [5]'s approach, independent windows are generated based on a minimum abnormal rate and repeat size.

This way, continuous monitoring samples will be generated, and training and testing windows will be sequentially extracted from these samples. Each sequence is presumed to start in a stable state without any assignable causes, beginning with k sampling windows in a stable condition from the data generated in the previous step. The value of k, randomly chosen between 6 and 17, aims to simulate the randomness of a fault and will also be used to calculate false alarm metrics. Following the k stable windows are 30 window samples of a specific abnormal pattern, making each sequence consist of k stable windows and 30 unnatural windows. There will be 9000 and 3600 sequences for training and testing each unnatural pattern, respectively. The grey columns in Table 3 detail the total number of windows (natural and unnatural) required for training and testing.

For example, the number of natural windows per pattern type for training is given by:

$$\sum_{k=6}^{17} (9000/12) * k = 103,500 \tag{8}$$

While the number of unnatural windows per pattern type for training is given by:

$$9000 * 30 = 270,000 \tag{9}$$

To ensure a fair comparative evaluation during the sensitivity analysis, the research team will conduct the training and testing processes over the same moving windows, thus maintaining consistency across different window sizes.

5. Results

As previously highlighted, sixteen patterns were tested across eight window sizes: 8, 16, 24, 32, 40, and 48, evaluating five hidden sizes: 30, 60, 120, 180, and 240. The same moving windows were maintained across each window size analysis. The findings, showcased in Figure 7, reveal that window sizes of 16 and 24 observations yield a high DA, in contrast to sizes 8, 32, 40, and 48, which demonstrated a low DA. Additionally, a larger window size necessitates a higher AUPRL, requiring more samples to detect unnatural variations, as suggested by Figure 7b,c. Lastly, Figure 7d illustrates that a smaller window size incurs a higher AFA than a larger one. These results emphasise the importance of adjusting window size and hidden parameters to enhance pattern detection efficiency.

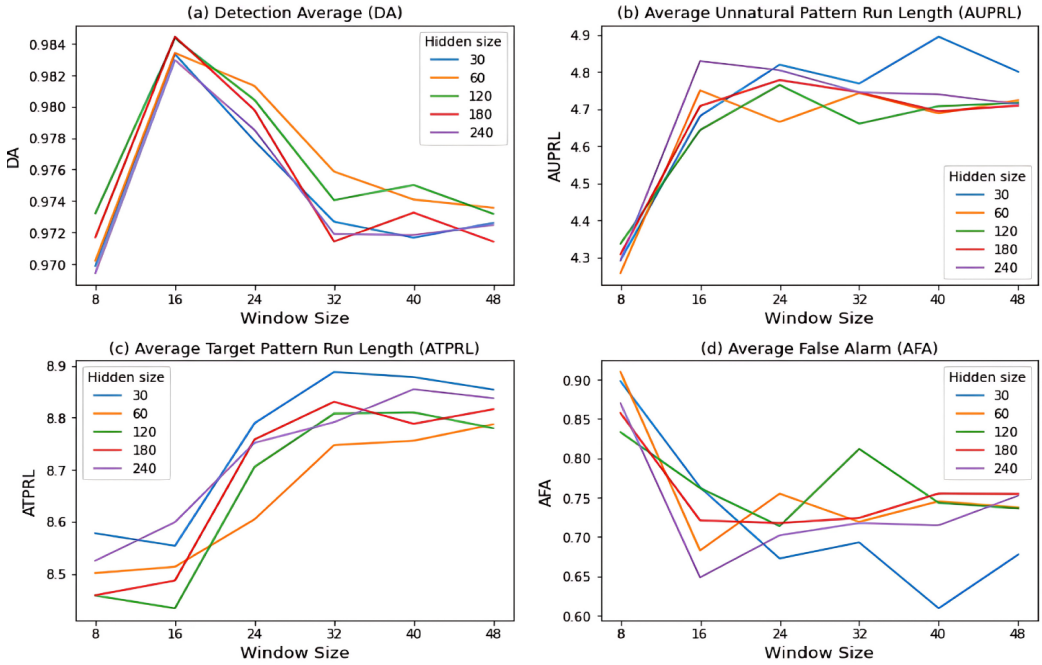


Figure 7. Window size sensitivity analysis. (**a**) detection average; (**b**) average unnatural pattern run length; (**c**) average target pattern run length; (**d**) average false alarm.

From the analysis in Figure 7, a performance trade-off is evident across the metrics considered: larger window sizes result in lower DA and AFA but higher AUPRL and ATPRL. Conversely, smaller window sizes exhibit lower ATPRL and higher AFA, indicating that large window sizes are less sensitive to the appearance of the unnatural pattern. This suggests that larger windows require more samples to detect a pattern while maintaining a lower rate of false alarms. This finding aligns with [45], who focused solely on the ARL metric, highlighting the complex balance between sensitivity to pattern changes and the minimisation of false alarms in pattern recognition tasks. This insight is crucial for optimising model parameters in real-world applications, balancing the need for accurate pattern detection against the practical limitations of false alarm rates.

The false alarm analysis shown in Figure 8 also confirms the previous result. Each colour in Figure 8 represents the average number of false alarms generated per unnatural pattern type for each window size. Figure 8a shows that once a false alarm has occurred, a large window size has more false alarms than a small window size, i.e., a large window has a higher NFA-NFAMW. However, Figure 8b shows that, during all moving windows analysis, there are fewer moving windows with at least one false alarm when a large size window is used, i.e., larger windows have a lower NFAMW-TMW. Furthermore,

Figure 8c shows that longer windows have a smaller NFA-TMW, i.e., they are less prone to false alarms.

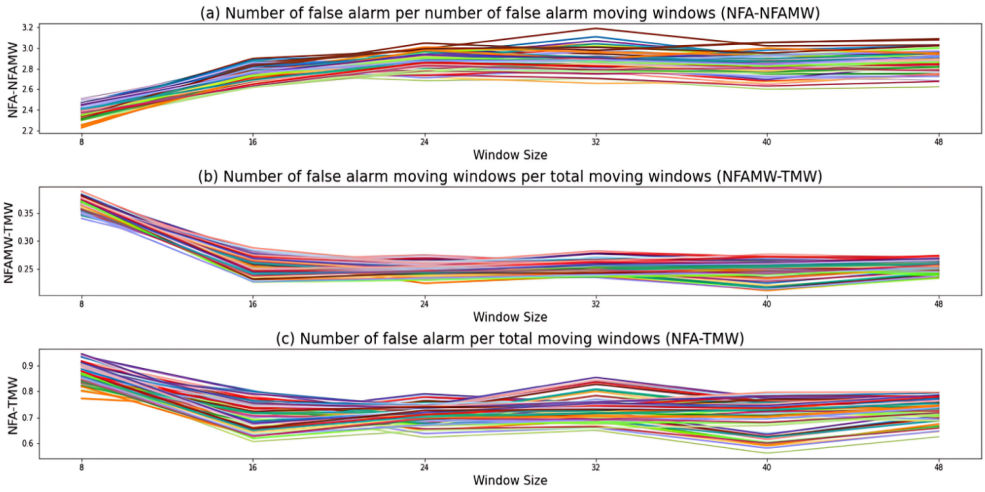


Figure 8. Summary of false alarm evaluation. (**a**) NFA-NFAMW; (**b**) NFAMW-TMW; (**c**) NFA-TMW.

Integrating the false alarm analysis with the previous findings emphasises the critical role of window size selection in balancing effective pattern detection and minimising false alarms. This balance is essential for addressing data complexity challenges and achieving optimal real-time detection. It highlights the importance of adjusting window sizes to improve operational efficiency and reduce downtime in advanced manufacturing environments. It directly connects with the need for scalable and robust models that provide an optimal balance between early detection and reduction in false alarms.

The hidden size analysis reveals that, unlike window size, this parameter does not show a clear pattern in its impact on performance metrics, indicating a less significant influence on model effectiveness (Figure 9). Results show that smaller hidden sizes lead to more extended training periods (Figure 10). Furthermore, the ratio of trainable model parameters to training windows suggests a delicate balance between the risks of overfitting and underfitting (Table 4). Increasing hidden size demands more training data to leverage the model's added complexity. These findings, suggesting a hidden size of 120 to balance training needs with model capacity, pave the way for a deeper discussion on how such technical decisions align with process optimisation goals in industrial environments.

Table 4. Summary of training data model vs. model size

	Hidden Size				
	30	**60**	**120**	**180**	**240**
Total parameters	4456	16,096	60,976	134,654	237,136
Total windows	5,602,500	5,602,500	5,602,500	5,602,500	5,602,500
Total windows/Total parameters	1257.29	348.07	91.88	41.61	23.63

Figure 11 shows a detailed summary of experimental ATPRL results per each unnatural pattern. It can be observed which patterns require a more significant number of samples to be detected, i.e., a higher ATPRL. Those patterns were TP, SP, CP-TP, TP-SP, and STP; on the other hand, the most easily detected were CP-SYP, SYP-SP, STP-SP, and STP-SYP patterns. Finally, the confusion matrix in Figure 12 shows which unnatural patterns were the most mistaken as natural patterns. Only unnatural patterns were considered during this analysis. Therefore, the natural pattern target row is null. It can be seen that the model tended to confuse the STP, TP, and STP-TP patterns with the natural pattern, i.e., a natural

variation was indicated when, in reality, there was an unnatural pattern. Furthermore, Figure 12 shows that the SYP-SP, STP-SP, SP, CP-SYP, and CP-SP patterns were the least confused as a natural pattern. These results are broadly consistent with the ATPRL analysis presented before.

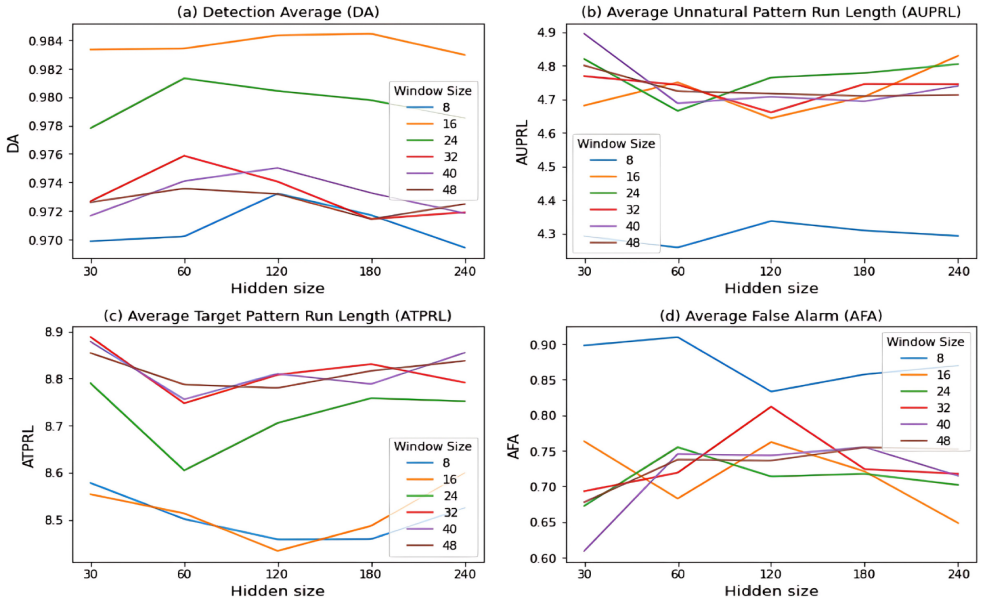


Figure 9. Hidden size sensitivity analysis. (**a**) detection average; (**b**) average unnatural pattern run length; (**c**) average target pattern run length; (**d**) average false alarm.

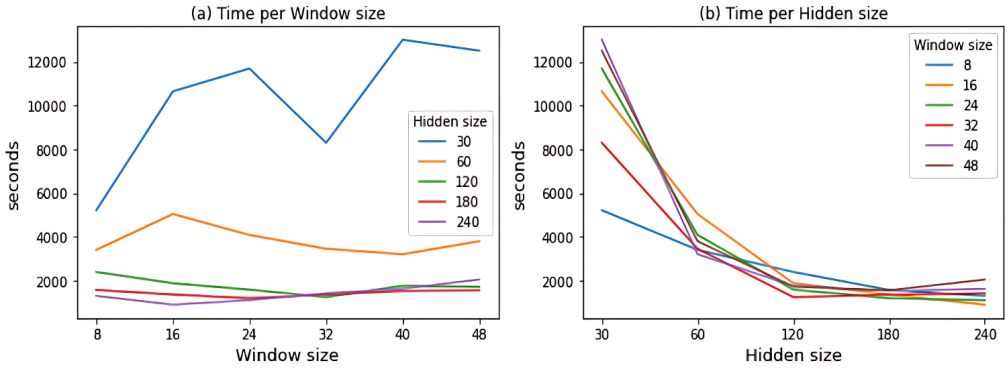


Figure 10. Summary of training time evaluation. (**a**) window size; (**b**) hidden size.

This study's findings contribute to understanding the dynamics of pattern recognition within classification models. By carefully analysing window sizes, hidden sizes, and pattern types, we have effectively demonstrated the model's nuanced ability to balance pattern detection against false alarm rates. Notably, selecting model parameters emerges as a critical factor in optimising performance. These insights fulfil this study's objectives and lay a solid foundation for further discussions on model optimisation and its implications in complex industrial settings.

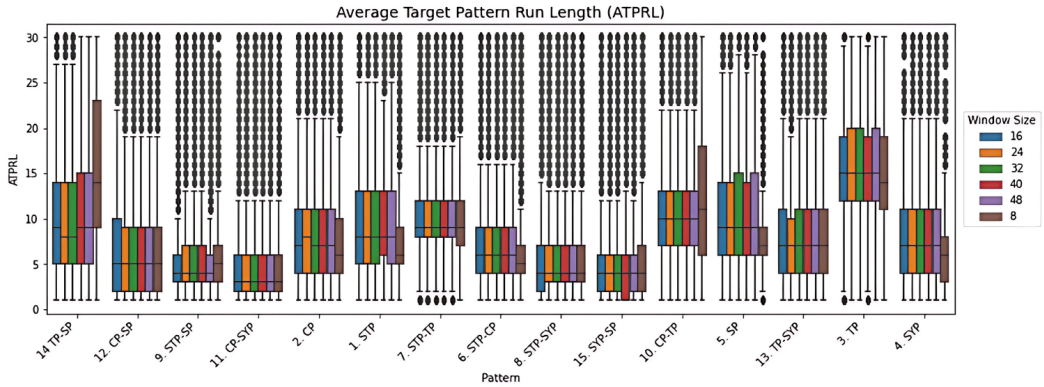


Figure 11. Summary of pattern identification performance.

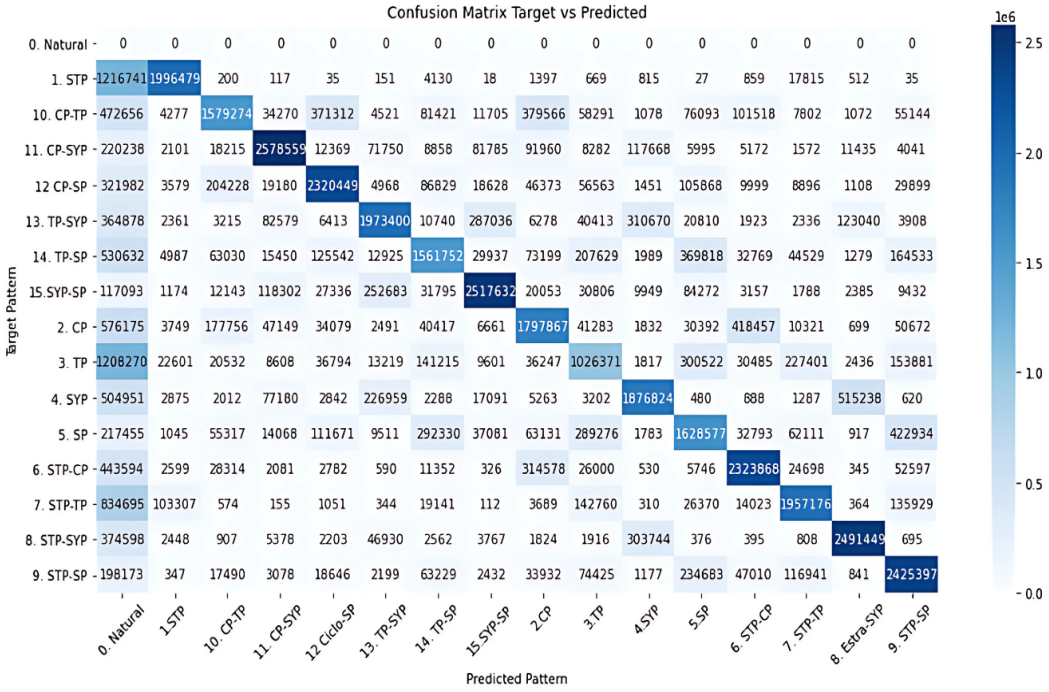


Figure 12. Confusion matrix for pattern analysis.

6. Discussion

Addressing contemporary challenges in concurrent pattern recognition within the modelling of real-world contexts demands research strategies and methods designed to overcome difficulties associated with data, pattern complexity, real-time detection capability, and achieving an optimal balance between early detection and false alarm reduction. Below, we explore how our study addresses these challenges, proposing solutions that significantly advance the optimisation of online detection and modelling of advanced industrial environments.

6.1. Addressing Data Challenges

At the forefront of sustainable manufacturing and the transition to Industry 4.0 and 5.0, we face inherent challenges in processing and analysing data to implement intelligent manufacturing systems. The challenges in pattern recognition presented by Fuqua and Razzaghi 2020 [6]—unbalanced data, noisy data, and large volumes of data—highlight the need to adapt new solution strategies to advanced manufacturing. Today, new technologies enable modern industries to achieve intelligent manufacturing systems. However, these advantages pose challenges and changes to traditional solution strategies. Emerging

technologies, such as the Internet of Things (IoT) and automatic process monitoring, provide tools for real-time process control but generate problems related to big data [46,47]. However, most studies for CCPR propose traditional ML models, although these models are not scalable for large volumes of data [6,48].

Moreover, in a real manufacturing environment, unnatural patterns are less common than natural ones, and consequently, unbalanced datasets could deteriorate the classification capability of a control system [6]. Thus, unbalanced data strategies should be implemented in future models. Furthermore, traditional ML models are not robust enough to handle noisy data, so tolerant strategies must be designed [6].

Strategies that directly address data imbalance, noise, and the ability to manage large-scale datasets must be adopted to move towards more sustainable and efficient manufacturing. This involves developing robust ML models that are tolerant to noisy data and implementing techniques that can effectively balance accuracy and generalisation in the presence of imbalanced data. Integrating these advanced approaches will improve classification and recognition capabilities in control systems and facilitate the transition to manufacturing practices essential for environmental sustainability and operational efficiency in the era of Industry 4.0 and 5.0.

To address the challenge of effectively managing data in the era of Industry 4.0 and 5.0, our study proposes a solution based on implementing long short-term memory (LSTM) neural networks. This approach stands out for its ability to process and analyse temporal data sequences, which is essential to identifying complex patterns and predicting failures in manufacturing processes. Using LSTM technology, the model learns from large volumes of historical data, adapting and improving its accuracy in predicting future events.

The obtained results demonstrate the capability of the designed network to be trained with more than 5.5 million data windows, each containing between 8 and 48 observations. This achievement overcomes the limitations of traditional models that do not take advantage of advanced technologies such as GPUs and training parallelisation. To counteract the imbalance between natural and unnatural patterns, we implemented an oversampling strategy, increasing the representation of unnatural patterns to improve their detection. However, it was observed that excessive oversampling can reduce the model's ability to identify natural patterns, increasing false alarms, while insufficient oversampling decreases its detection capability.

This model proposes a strategy for solving big data issues in modern industrial environments. It also drives informed, real-time decision-making, improves operational efficiency, and reduces machine and process downtime. The proposed solution represents a move towards Industry 4.0 and 5.0 data management, providing a robust framework for process optimisation and continuous improvement in the manufacturing sector.

Adopting these strategies involves a shift towards how we approach problem-solving in intelligent manufacturing, highlighting the importance of close collaboration between data engineering, data science, and sustainability. This integrative approach is critical to overcoming current challenges and unlocking the full potential of intelligent manufacturing towards a more sustainable future.

6.2. Complexity

The Industry 4.0 paradigm, and progressively that of Industry 5.0, requires scalable machine learning (ML) models. However, most current approaches are not designed to accommodate the scale and complexity of data generated in intelligent manufacturing environments [6,48]. This limitation is particularly critical when considering the concurrency of multiple patterns in complex and automated production processes, necessitating the search for innovative solutions that can handle these complexities [9,41]. The concurrence of

multiple patterns is joint in complex automated production processes [7,49]. The transition to innovative manufacturing systems drives the need for advanced monitoring systems capable of integrating concurrent patterns and challenges researchers to create advanced monitoring systems that integrate these concurrent patterns, preventing deterioration in the effectiveness of current monitoring models. This need is further emphasised by research demonstrating how the omission of concurrent patterns can negatively affect the performance of these systems [50,51].

Therefore, feature extraction is fundamental to improving concurrent pattern recognition, and many authors show how this approach increases pattern identification performance [2,49,51]. However, manually defined feature extraction is a strenuous task because these features must be designed appropriately based on prior knowledge of the pattern features, and poor feature selection could quickly deteriorate model performance [11,25].

To effectively address complexity and improve scalability in the era of Industry 4.0 and 5.0, ML researchers and system developers need to evolve their approaches to methods that can effectively handle the challenges presented by pattern complexity. This includes innovation in feature extraction techniques that minimises manual intervention and maximises models' ability to autonomously learn the complexities inherent in real smart manufacturing data. Overcoming these barriers will lead to greater efficiency and accuracy in pattern recognition and facilitate a smoother transition to highly advanced and sustainable manufacturing systems.

Unlike traditional machine learning models, the LSTM model proposed in this study does not require manual feature extraction to improve the detection process. Moreover, complex features resulting from pattern concurrence can be extracted by applying the LSTM network without separating the concurrent patterns into the simple patterns that comprise it, as proposed by numerous monitoring strategies. The model performance evaluation exposed the model's ability to detect concurrent patterns, especially for the CP-SYP, SYP-SP, STP- SYP, STP-CP, CP-SP, and STP-SP patterns (Figure 11). However, a strategy with multiple classifier modules might be desirable to improve model performance. This strategy could include a module with a binary classification output, whether the pattern should go to a specific classifier or not, and in the second stage, specialised classifiers could discriminate the pattern class.

Furthermore, the confusion matrix (Figure 12) showed that the unnatural patterns STP, TP, and their combination were easily confused with the natural pattern, resulting in more false negatives than the other included patterns. This may be due to the small magnitudes of change applied to the STP and TP and their combination concerning the natural pattern. Thus, the proposed multi-stage classification could be constructed by designing specialised classifiers, one for detecting small and large changes and a binary module to split the multi-stage classification.

6.3. Balancing Early Detection and False Alarms

Optimising the balance between early anomaly detection and minimisation of false alarms is challenging in developing process control (PC) models. This balance is relevant since a high sensitivity in anomaly detection can lead to an increase in alarm signals without the presence of an actual disruptive factor, which is counterproductive in practical application contexts. On the other hand, a model that generates few false alarms but identifies anomalies late also needs more practical value, as it compromises the ability to take corrective actions in time.

In this regard, it is critical to design and implement models that are highly sensitive to actual failures and maintain a low detection error rate. This implies a meticulous approach to feature selection, parameter calibration, and the design of algorithms that can effectively

discern between normal operating conditions and incipient anomalous events. Adopting advanced machine learning and data analysis techniques, such as deep learning and signal processing, can significantly improve the early detection of CCPR models, enabling more accurate identification of anomalies without increasing false alarms.

To achieve this balance, it is essential to implement validation and testing strategies that simulate real operational scenarios, thus allowing for adjustment of the model's sensitivity and ability to discriminate between normal fluctuations and significant changes that require attention. In addition, integrating artificial intelligence techniques that learn continuously and adaptively can facilitate the progressive improvement of the model based on operational feedback, refining its accuracy in detecting failures and reducing incidences of false alarms.

In the framework of Industry 4.0 and 5.0, where efficiency and sustainability are key objectives, the ability of CCPR models to effectively balance early anomaly detection with minimising false alarms is a necessary component. This ensures greater continuity and efficiency in production processes and reduces operating costs and promotes sustainable manufacturing practices. Ultimately, achieving this balance improves companies' resilience and competitiveness in today's dynamic industrial environment.

For accurate evaluation in real-time or online application scenarios, models should be evaluated with metrics that reflect early detection, accuracy, and false alarm incidence. Under this precedent, it was proposed to implement seven (7) different metrics to measure model performance comprehensively. The importance of these metrics lies in the fact that the prediction of the current state of the process can reflect three different conditions: the first is to correctly indicate the state of the process, either in control or out of control; the second is to indicate the existence of a fault when it does not exist, indicating a false alarm; and the third is to indicate the absence of a fault when the process is out of control. Each of these conditions is associated with a different cost in terms of operation; false alarms generate costs associated with stopping the process when there is no assignable cause, and not signalling the existence of faults implies costs of producing products out of ideal conditions. Accuracy cannot signal what type of condition is present in the prediction alone, limiting the process of practical evaluation of the model's performance. In addition, it does not provide information on the early detection of failures, i.e., how quickly the model detects the presence of a disturbing factor.

The study of the trade-off between false alarms and early detection is proposed to adequately evaluate the model performance. It was found that for the proposed model, increasing early detection tends to increase false alarms. Strategies such as adjusting the window size or the data ratio during training are crucial to managing this trade-off. It is vital to evaluate these strategies in real contexts to minimise operational costs, considering that false alarms may cost more than no fault detection, depending on the operating conditions and process objectives.

6.4. Improve Online Detection Capability

The implementation of real-time detection systems is required in practical applications of industrial processes, where the primary purpose lies in the early or timely identification of assignable causes through the analysis of successive samples to discern the current state of the process. This monitoring strategy increases the complexity of pattern recognition since it promotes the gradual appearance of the atypical pattern within the sampled data, challenging its effective identification.

The existing literature on concurrent pattern recognition in process control (CCPR) [7,11] often assumes a fully visualised pattern rather than a gradual emergence during model design (see Figure 1), limiting its applicability in real process control scenarios. Thus,

it highlights the need for innovation in data simulation methodologies that enhance early detection capabilities in future CCPR models.

In the context of Industry 4.0 and 5.0, where the vision of zero-defect manufacturing and sustainability is positioned at the epicentre of industrial practices, the need to move towards more sophisticated and efficient monitoring and pattern recognition systems is more relevant than ever. Implementing CCPR models that not only accurately anticipate the emergence of anomalies but also minimise false alarms becomes an indispensable requirement. This evolution towards more accurate and less intrusive detection systems is essential to achieve more efficient manufacturing, reduce resource waste, and promote more sustainable production practices. By addressing these challenges with technological and methodological innovations, the industry can move towards a future where operational efficiency and environmental sustainability can be achieved harmoniously with the environment.

To optimise online detection, this study introduces a simulation model that randomly generates a fault between 6 and 17 natural samples, seeking to replicate the spontaneous occurrence of anomalous patterns. Thirty post-fault samples are added to facilitate pattern identification by the model. In contrast to previous studies, seven metrics are used instead of only accuracy for a comprehensive evaluation, including the analysis of the trade-off between false alarms and early detection, influenced by window size and network parameters. Variations in window size are found to affect model sensitivity, but the challenge of balancing both characteristics remains open. Similarly, the hidden size does not significantly impact the metrics, although its optimisation is crucial to balance training time and memory requirements.

Likewise, this study improves online detection capability by simultaneously addressing critical challenges such as handling complex data, balancing early detection, minimising false alarms, and the complexity of concurrent patterns. An improvement in real-time detection capability is demonstrated by implementing an appropriate simulation model and utilising multiple metrics for a more comprehensive evaluation. This comprehensive approach, supported by the use of LSTM technology, not only overcomes the limitations of traditional models but also establishes a framework for the future development of more efficient and accurate monitoring systems in the context of Industry 4.0 and 5.0.

7. Conclusions

This study introduces a novel evaluation methodology for pattern recognition in advanced quality control. By implementing the proposed structure, we have estimated the model's sensitivity to variations in window size and hidden layer size and analysed the conditions that best maintain a balance between early fault detection and false alarm minimisation. The results demonstrated that the LSTM model can detect faults during the evaluation process by dynamically adapting to the temporal variations inherent in the simulated manufacturing processes.

However, evaluating model performance under different metrics and parameters presents the limitation of defining a general performance indicator that enables a global comparison of monitoring proposals, as authors currently do. One strategy to overcome this limitation could involve comparing the trade-off between false alarms and early detection to select the model with the best overall performance. In the evaluated case, experimental evaluation reveals that the optimal window size effectively balances model sensitivity and false alarm minimisation, with an approximate average size of 24 observations emerging as ideal. Comparing models under this optimal balance can help determine which model demonstrates superior overall performance. Although the hidden layer size setting marginalises model performance, larger configurations were observed to optimise training time.

Our findings underscore the complexity of identifying specific anomaly patterns, highlighting particularly challenging TP, SP, CP-TP, and STP patterns. In contrast, patterns such as CP-SYP, SYP-SP, STP-SP, CP-SP, and STP-SYP proved to be more distinguishable. The research concludes by emphasising the need to create models focused on the current needs of the industry, emphasising the increasing automation of processes and the emergence of new research challenges in CCPR. This study not only validates the versatility and effectiveness of a classification model in advanced manufacturing environments but also sets the path for future research that can explore the sensitivity of a particular classification model in increasingly complex and dynamic industrial scenarios.

Author Contributions: Conceptualization, E.G.; methodology, E.G.; software, E.G.; validation, E.G. and K.P.; formal analysis, E.G., R.P.-N., W.S.P. and K.P.; investigation, E.G. and K.P.; resources, E.G.; data curation, E.G. and K.P.; writing—original draft preparation, E.G. and K.P.; writing—review and editing, R.P.-N. and W.S.P.; visualization, E.G. and K.P.; supervision, R.P.-N. and W.S.P.; project administration, R.P.-N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SPC	Statistical process control
CCPR	Control chart pattern recognition
ANN	Artificial neural network
DT	Decision tree
SVM	Support vector machine
DNN	Deep neural networks
LSTM	Long short-term memory
RNN	Recurrent neural network
FC	Fully connected layer
ARL	Average run length
ATPRL	Average target pattern run length
AUPRL	Average unnatural pattern run length
DA	Detection average
AFA	Average false alarm
NFA	Number of false alarms
NFAMW	Number of moving windows with at least one false alarm
TMW	Total number of moving windows
HN	Hidden network sizes
TP	Trend pattern
SP	Shift pattern
CP	Cycle pattern
STP	Stratified pattern
SYS	Systematic pattern

References

1. Goujon, A.; Rosin, F.; Magnani, F.; Lamouri, S.; Pellerin, R.; Joblot, L. Industry 5.0 use cases development framework. *Int. J. Prod. Res.* **2024**, *62*, 6064–6089. [CrossRef]
2. Pelegrina, G.D.; Duarte, L.T.; Jutten, C. Blind source separation and feature extraction in concurrent control charts pattern recognition: Novel analyses and a comparison of different methods. *Comput. Ind. Eng.* **2016**, *92*, 105–114. [CrossRef]
3. De la Torre-Gutiérrez, H.; Pham, D. A control chart pattern recognition system for feedback-control processes. *Expert Syst. Appl.* **2019**, *138*, 112826. [CrossRef]
4. Lu, Z.; Wang, M.; Dai, W. A condition monitoring approach for machining process based on control chart pattern recognition with dynamically-sized observation windows. *Comput. Ind. Eng.* **2020**, *142*, 106360. [CrossRef]
5. Ünlü, R. A robust data simulation technique to improve early detection performance of a classifier in control chart pattern recognition systems. *Inf. Sci.* **2021**, *548*, 18–36. [CrossRef]
6. Fuqua, D.; Razzaghi, T. A cost-sensitive convolution neural network learning for control chart pattern recognition. *Expert Syst. Appl.* **2020**, *150*, 113275. [CrossRef]
7. Chiu, J.E.; Tsai, C.H. On-line concurrent control chart pattern recognition using singular spectrum analysis and random forest. *Comput. Ind. Eng.* **2021**, *159*, 107538. [CrossRef]
8. Yang, W.A.; Yu, G.; Liao, W. A Hybrid Learning-Based Model for Simultaneous Monitoring of Process Mean and Variance. *Qual. Reliab. Eng. Int.* **2015**, *31*, 445–463. [CrossRef]
9. Lesany, S.; Ghomi, S.F.; Koochakzadeh, A. Development of fitted line and fitted cosine curve for recognition and analysis of unnatural patterns in process control charts. *Pattern Anal. Appl.* **2019**, *22*, 747–765. [CrossRef]
10. Kalteh, A.A.; Babouei, S. Control chart patterns recognition using ANFIS with new training algorithm and intelligent utilization of shape and statistical features. *ISA Trans.* **2020**, *102*, 12–22. [CrossRef]
11. Ünlü, R. Cost-Oriented LSTM Methods For Possible Expansion of Control Charting Signals. *Comput. Ind. Eng.* **2021**, *154*, 107163. [CrossRef]
12. Hachicha, W.; Ghorbel, A. A survey of control-chart pattern-recognition literature (1991–2010) based on a new conceptual classification scheme. *Comput. Ind. Eng.* **2012**, *63*, 204–222. [CrossRef]
13. Addeh, A.; Khormali, A.; Golilarz, N.A. Control chart pattern recognition using RBF neural network with new training algorithm and practical features. *ISA Trans.* **2018**, *79*, 202–216. [CrossRef] [PubMed]
14. Guh, R.S.; Hsieh, Y.C. A neural network based model for abnormal pattern recognition of control charts. *Comput. Ind. Eng.* **1999**, *36*, 97–108. [CrossRef]
15. Perry, M.B.; Spoerre, J.K.; Velasco, T. Control chart pattern recognition using back propagation artificial neural networks. *Int. J. Prod. Res.* **2001**, *39*, 3399–3418. [CrossRef]
16. Al-Assaf, Y. Recognition of control chart patterns using multi-resolution wavelets analysis and neural networks. *Comput. Ind. Eng.* **2004**, *47*, 17–29. [CrossRef]
17. Ebrahimzadeh, A.; Ranaee, V. Control chart pattern recognition using an optimized neural network and efficient features. *ISA Trans.* **2010**, *49*, 387–393. [CrossRef]
18. Yang, W.A.; Zhou, W. Autoregressive coefficient-invariant control chart pattern recognition in autocorrelated manufacturing processes using neural network ensemble. *J. Intell. Manuf.* **2015**, *26*, 1161–1180. [CrossRef]
19. Guh, R.S.; Shiue, Y.R. An effective application of decision tree learning for on-line detection of mean shifts in multivariate control charts. *Comput. Ind. Eng.* **2008**, *55*, 475–493. [CrossRef]
20. Wang, C.H.; Guo, R.S.; Chiang, M.H.; Wong, J.Y. Decision tree based control chart pattern recognition. *Int. J. Prod. Res.* **2008**, *46*, 4889–4901. [CrossRef]
21. Wu, S.X.; Huang, E.Z. Control chart patterns recognition based on support vector machine. *Jisuanji Yingyong/J. Comput. Appl.* **2007**, *27*, 61–64.
22. Lin, S.Y.; Guh, R.S.; Shiue, Y.R. Effective recognition of control chart patterns in autocorrelated data using a support vector machine based approach. *Comput. Ind. Eng.* **2011**, *61*, 1123–1134. [CrossRef]
23. Xanthopoulos, P.; Razzaghi, T. A weighted support vector machine method for control chart pattern recognition. *Comput. Ind. Eng.* **2014**, *70*, 134–149. [CrossRef]
24. Kao, L.J.; Lee, T.S.; Lu, C.J. A multi-stage control chart pattern recognition scheme based on independent component analysis and support vector machine. *J. Intell. Manuf.* **2016**, *27*, 653–664. [CrossRef]
25. Golilarz, N.A.; Addeh, A.; Gao, H.; Ali, L.; Roshandeh, A.M.; Munir, H.M.; Khan, R.U. A new automatic method for control chart patterns recognition based on ConvNet and harris hawks meta heuristic optimization algorithm. *IEEE Access* **2019**, *7*, 149398–149405. [CrossRef]
26. Wang, C.H.; Kuo, W. Identification of control chart patterns using wavelet filtering and robust fuzzy clustering. *J. Intell. Manuf.* **2007**, *18*, 343–350. [CrossRef]

27. Ebrahimzadeh, A.; Addeh, J.; Rahmani, Z. Control chart pattern recognition using K-MICA clustering and neural networks. *ISA Trans.* **2012**, *51*, 111–119. [CrossRef]
28. Haghtalab, S.; Xanthopoulos, P.; Madani, K. A robust unsupervised consensus control chart pattern recognition framework. *Expert Syst. Appl.* **2015**, *42*, 6767–6776. [CrossRef]
29. Khormali, A.; Addeh, J. A novel approach for recognition of control chart patterns: Type-2 fuzzy clustering optimized support vector machine. *ISA Trans.* **2016**, *63*, 256–264. [CrossRef]
30. Zaman, M.; Hassan, A. Improved statistical features-based control chart patterns recognition using ANFIS with fuzzy clustering. *Neural Comput. Appl.* **2019**, *31*, 5935–5949. [CrossRef]
31. Guh, R.S. Real-time pattern recognition in statistical process control: A hybrid neural network/decision tree-based approach. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2005**, *219*, 283–298. [CrossRef]
32. Shao, X. Recognition of control chart patterns using decision tree of multi-class SVM. In *Advances in Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 33–41.
33. Wu, C.; Liu, F.; Zhu, B. Control chart pattern recognition using an integrated model based on binary-tree support vector machine. *Int. J. Prod. Res.* **2015**, *53*, 2026–2040. [CrossRef]
34. Yu, J. Gaussian mixture models-based control chart pattern recognition. *Int. J. Prod. Res.* **2012**, *50*, 6746–6762. [CrossRef]
35. Shao, Y.E.; Chiu, C.C. Applying emerging soft computing approaches to control chart pattern recognition for an SPC–EPC process. *Neurocomputing* **2016**, *201*, 19–28. [CrossRef]
36. Bo, Z.; Beibei, L.; Yuwei, W.; Shengran, Z. Recognition of control chart patterns in auto-correlated process based on random forest. In Proceedings of the 2018 IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE), Hsinchu, Taiwan, 8–9 February 2018; pp. 53–57.
37. Chen, L.; Xu, H.; Lan, K.; Zheng, Y. Control Chart Pattern Recognition of Sheet Metal Cutting Data in Shipbuilding Based on XGBoost. In Proceedings of the 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), Hong Kong, China, 20–21 August 2020; pp. 500–507.
38. Xue, L.; Wu, H.; Zheng, H.; He, Z. Control chart pattern recognition for imbalanced data based on multi-feature fusion using convolutional neural network. *Comput. Ind. Eng.* **2023**, *182*, 109410. [CrossRef]
39. Derakhshi, M.; Razzaghi, T. An imbalance-aware BiLSTM for control chart patterns early detection. *Expert Syst. Appl.* **2024**, *249*, 123682. [CrossRef]
40. García, E.; Peñabaena-Niebles, R.; Jubiz-Diaz, M.; Perez-Tafur, A. Concurrent Control Chart Pattern Recognition: A Systematic Review. *Mathematics* **2022**, *10*, 934. [CrossRef]
41. Guh, R.S.; Tannock, J. Recognition of control chart concurrent patterns using a neural network approach. *Int. J. Prod. Res.* **1999**, *37*, 1743–1765. [CrossRef]
42. Zan, T.; Liu, Z.; Su, Z.; Wang, M.; Gao, X.; Chen, D. Statistical process control with intelligence based on the deep learning model. *Appl. Sci.* **2020**, *10*, 308. [CrossRef]
43. Kao, L.J.; Chiu, C.C. Application of integrated recurrent neural network with multivariate adaptive regression splines on SPC-EPC process. *J. Manuf. Syst.* **2020**, *57*, 109–118. [CrossRef]
44. Yu, Y.; Zhang, M. Control Chart Recognition Based on the Parallel Model of CNN and LSTM with GA Optimization. *Expert Syst. Appl.* **2021**, *185*, 115689. [CrossRef]
45. Chen, S.; Yu, J. Deep recurrent neural network-based residual control chart for autocorrelated processes. *Qual. Reliab. Eng. Int.* **2019**, *35*, 2687–2708. [CrossRef]
46. Shang, C.; You, F. Data analytics and machine learning for smart process manufacturing: Recent advances and perspectives in the big data era. *Engineering* **2019**, *5*, 1010–1016. [CrossRef]
47. Xu, J.; Lv, H.; Zhuang, Z.; Lu, Z.; Zou, D.; Qin, W. Control Chart Pattern Recognition Method Based on Improved One-dimensional Convolutional Neural Network. *IFAC-PapersOnLine* **2019**, *52*, 1537–1542. [CrossRef]
48. Shrestha, A.; Mahmood, A. Review of deep learning algorithms and architectures. *IEEE Access* **2019**, *7*, 53040–53065. [CrossRef]
49. Zhang, M.; Zhang, X.; Wang, H.; Xiong, G.; Cheng, W. Features Fusion Exaction and KELM with Modified Grey Wolf Optimizer for Mixture Control Chart Patterns Recognition. *IEEE Access* **2020**, *8*, 42469–42480. [CrossRef]
50. Xie, L.; Gu, N.; Li, D.; Cao, Z.; Tan, M.; Nahavandi, S. Concurrent control chart patterns recognition with singular spectrum analysis and support vector machine. *Comput. Ind. Eng.* **2013**, *64*, 280–289. [CrossRef]
51. Zhang, M.; Yuan, Y.; Wang, R.; Cheng, W. Recognition of mixture control chart patterns based on fusion feature reduction and fireworks algorithm-optimized MSVM. *Pattern Anal. Appl.* **2020**, *23*, 15–26. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.