Fault detection and diagnosis to enhance safety in digitalized process system

Kopbayev, Alibek; Khan, Faisal; Yang, Ming; Halim, S. Zohra

DOI
10.1016/j.compchemeng.2021.107609

Publication date
2022

Document Version
Final published version

Published in
Computers and Chemical Engineering

Computers and Chemical Engineering 158 (2022) 107609

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Fault detection and diagnosis to enhance safety in digitalized process system

Alibek Kopbayev[a], Faisal Khan[b,*], Ming Yang[c], S. Zohra Halim[b]

[a] Centre for Risk, Integrity and Safety Engineering (C-RISE), Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's A1B 3×5, NL, Canada
[b] Mary Kay O'Connor Process Safety Center, Artie McFerrin Department of Chemical Engineering, Texas A&M University, College Station, TX, USA
[c] Safety and Security Science Section, Faculty of Technology, Policy and Management, Delft University of Technology, The Netherlands

ARTICLE INFO

Article history:
Received 4 July 2021
Revised 18 November 2021
Accepted 19 November 2021
Available online 22 November 2021

Keywords:
Deep Neural Networks
kPCA
Fault detection and diagnosis
Process system safety
Hybrid model

ABSTRACT

The increased complexity of digitalized process systems requires advanced tools to detect and diagnose faults early to maintain safe operations. This study proposed a hybrid model that consists of Kernel Principal Component Analysis (kPCA) and DNNs that can be applied to detect and diagnose faults in various processes. The complex data is processed by kPCA to reduce its dimensionality; then, simplified data is used for two separate DNNs for training (detection and diagnosis). The relative performance of the hybrid model is compared with conventional methods. Tennessee Eastman Process was used to confirm the efficacy of the model. The results show the reduction of input dimensionality increases classification accuracy. In addition, splitting detection and diagnosis into two DNNs results in reduced training times and increased classification accuracy. The proposed hybrid model serves as an important tool to detect the fault and take early corrective actions, thus enhancing process safety.

1. Introduction

Modern engineering systems have become more sophisticated and complex due to increased interactions between process sensors and actuators to pursue optimization and automation. The actuators control critical parameters of the system while the sensors keep track of any changes in the system. In some cases, the process fails to operate within safe boundaries due to various faults. An unsafe operation may lead to equipment or system failure and can eventually result in an accident. To prevent this from happening, fault detection and diagnosis methods have been proposed to ensure a safe process environment and to maintain product quality (Venkatasubramanian et al., 2003b).

Fault detection and diagnosis is a monitoring system that is aimed to identify deviations in the system or its parameters. As such, early identification is then followed by corrective measurements that result in accident prevention or damage mitigation. Fault detection and diagnosis can be generalized into quantitative/analytical model-based methods, qualitative model-based methods, and data-driven methods (Venkatasubramanian et al., 2003b). Quantitative model-based methods include techniques such as regression parameter estimation (Wu and Liu, 2017), least-squares parameter estimation (Cimpoesu et al., 2013; Isermann, 1993), linear quadratic estimation (Amoozgar et al., 2013; Huang et al., 2012), and others. Qualitative based approaches have also been thoroughly investigated, such as fault tree (Antonio et al., 1995), functional abstraction (Ham and Yoon, 2001), structural hierarchy (Lind, 1999), fuzzy logic systems (Nan et al., 2008), directed graph-based methods (Gao et al., 2010). Data-driven methods rely on process data to identify operational conditions and detect abnormal behaviors. Feature extraction is the crucial step in data-driven methods, which focuses on condensing the process data into more practical information. Univariate and multivariate approaches have been deployed to perform data transformation. Several multivariate methods have been successful in overcoming shortcomings of traditional approaches, such as dynamic principal component analysis (Ku et al., 1995), independent component analysis (Kano et al., 2003), modified partial least squares (Yin et al., 2011), and others. Recent years have seen some studies that attempt to link the qualitative-based approaches to the data-driven approaches (Sarbayev et al., 2019).

As engineering processes get more complex, analytical models cannot consider the increasing number of highly correlated dy-

* Corresponding author at: Mary Kay O'Connor Process Safety Center, Artie McFerrin Department of Chemical Engineering, Texas A&M University, College Station, TX, USA
E-mail addresses: fikhan@mun.ca, fikhan@tamu.edu (F. Khan).

https://doi.org/10.1016/j.compchemeng.2021.107609
0098-1354/

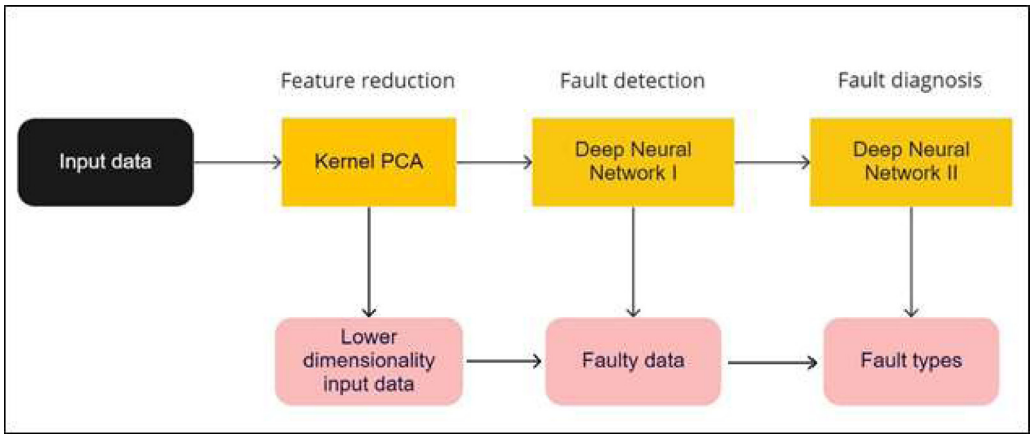

Fig. 1. Methodology to develop a hybrid model for fault detection and diagnosis using deep neural networks.

namic system interactions (Ge et al., 2013; Venkatasubramanian et al., 2003a). A recent direction in a data-driven approach has focused on applying neural networks in fault detection and diagnosis tasks. Neural networks showed successful application in complex classification tasks in various fields, such as speech recognition (Hinton et al., 2012), digit recognition (Kayumov et al., 2020), image recognition (Simonyan and Zisserman, 2015), and others. It was proposed that neural networks have the potential for fault detection and diagnosis (Zhang and Zhao, 2017). This is possible because neural networks consist of a structured network of neurons that can learn sophisticated patterns via nonlinear transformations. A faulty operation can be treated as a specific pattern in process data, which a trained neural network will detect.

Modified neural networks have been successfully applied to perform fault detection and/or diagnosis (Heo and Lee, 2018; Ince et al., 2016; Su et al., 2020; Tang et al., 2019; Wen et al., 2018) of various processes ranging from Tennessee Eastman process to motor lifetime estimation. However, most of these works dealt with low-dimensionality data, which is suitable for neural networks. When the complexity of data increased, the computations became time expensive and inapplicable to complex equipment/systems. In addition, Tang et al. (2019) suggested that hybrid models that combine data preprocessing models with neural networks might increase the detection accuracy, but it has not been tested before. Finally, limited work has been done to support dynamic data, and issues of dealing with increase in the dimensions of the data matrix and interpretability of the data analytics are a common challenge among them (Dong and Joe Qin, 2018).

This work proposes a hybrid method that deals with high-dimensionality data and performs fault detection and diagnosis. Kernel Principal Component Analysis (kPCA) was used as the first step of our model: this can process nonlinear data and reduce its dimensionality with minimal loss of variability. Then, we trained a deep neural network (DNN) for fault classification as the second step. We separated fault detection and diagnosis into consecutive deep neural networks to simplify the tuning process and reduce training time. Although both kPCA and DNN were used in literature for fault detection and diagnosis purposes (Maki and Loparo, 1997; Navi et al., 2018), a combination of both has not yet been explored.

We used the digit classification dataset as a substitute due to its high dimensionality, which allowed us to monitor classification accuracy for different model configurations. Then, we applied the hybrid model to the Tennessee Eastman Process (TEP) to confirm the effectiveness in detecting and diagnosing various process faults.

The remaining part of this paper is organized as follows. Section 2 presents an overview of the hybrid model and provides a brief background on kPCA, fault detection and diagnosis, and Deep Neural Networks. The performance of the proposed hybrid model is discussed via a case study in Section 3. Section 4 contains an analysis and discussion of the results of the case studies. Finally, Section 5 concludes the work and gives recommendations for future work.

2. Methodology to develop the hybrid model

2.1. Process overview

Fig. 1 shows that the process consists of three steps: feature reduction using kPCA, fault detection and fault diagnosis using two Deep Neural Networks (DNN) N_I and N_{II} respectively. The kPCA step is used as a feature reduction step that transforms the dataset so that the neural network can process more accurately. The first neural network N_I distinguishes faults from normal operation, and the second neural network N_{II} differentiates the faults by nature. The resulting hybrid model can detect and diagnose faults from complex datasets. Fault detection and fault diagnosis can be conducted simultaneously or separately. Simultaneous detection and diagnosis are a classification task that is efficient in some situations. Only one neural network needs to be trained in that case, and therefore the training time is reduced, and all data is used for diagnosis without loss. However, the accuracy of such a model is highly dependent on the dataset. If a dataset is dominated by one category, there is a high chance of misclassification towards that class. In fault detection and diagnosis, this will usually be the issue since we are interested in the minority (faulty operation). The overall accuracy of the model will be high, but so will the number of false positives. If the data is evenly distributed between operational conditions and different types of faults, the performance of this network will be acceptable. However, this is not the usual case in industrial processes. Therefore, we split fault detection and fault diagnosis into separate classification models using two separate DNN for each.

2.2. Step 1. Reducing the dimensionality of input data with kernel principal component analysis (kPCA)

This technique aims to reduce the complexity of the input dataset to be used for the training of neural networks by reducing its dimensionality. Principal Component Analysis is commonly used for this purpose. It is an effective data modeling tool that can extract latent variables from complex datasets while maximizing the data variation. However, the basic PCA cannot efficiently separate nonlinear data. In this work, an extension of PCA, called kPCA, will

be used to handle linearly inseparable datasets. This method uses the integral operator kernel function to transform data in higher-feature space and then perform PCA Schölkopf et al., 1998) This is possible due to a "kernel trick", which allows us to avoid calculations in the feature space and to use dot product between points to find principal components Eqs. (1)–(12).

First, we map data into a higher feature space ϕ using Eq. (1).

$$X(i) \rightarrow \Phi(X(i)) \tag{1}$$

Then, the covariance matrix is transformed into Eq. (2).

$$C = \frac{1}{N}\sum_{i=1}^{N} \Phi(X(i))\Phi(X(i))^T \tag{2}$$

By denoting a dot product between datapoints in the mapped space as K (elements of the kernel), we perform eigendecomposition for kPCA Eq. (3).

$$N\lambda a = Ka \tag{3}$$

Where N is the number of observations, λ is eigenvalues, and a is eigenvectors.

However, we must first "centralize" the kernel Eq. (4) $K \rightarrow K'$

$$K' = K - 1_N K - K1_N + 1_N K 1_N \tag{4}$$

1_N is defined in Eq. (5):

$$1_N = \frac{1}{N} \tag{5}$$

Now, centralized kernel K' can be used to perform kPCA, and similar to basic PCA, we can use the eigenvalues to determine the variability of each principal component. Some of the widely used kernels are polynomial, radial basis function, sigmoid, Gaussian, exponential, and others (Zhang, 2015). There is a trial-and-error procedure of trying different kernel functions that can affect the performance of kPCA.

Challenges of kPCA.

The major issue with kPCA is that the kernel K is a N-by-N matrix, and for large datasets, this step may require a lot of resources (RAM and CPU). This method does not allow for dynamic analysis, as the addition of any data requires the calculation of kernel matrix from scratch.

In practice, a 16GB RAM computer can handle a dataset containing roughly 30,000 data points. For datasets beyond this size, modifications to kPCA are required. As for dynamic datasets, there exist extensions of kPCA, such as TP-IKPCA (Zhao et al., 2019), which can handle both large and incremental datasets.

2.3. Step 2. Classifying input data as faulty or faultless, i.e., fault detection

Fault detection is an integral part of improving the reliability and safety of a system. In this paper, we train a Deep Neural Network (DNN) to detect faults among process data. This is an example of supervised learning as it requires labeling of process data. We treat fault detection as a simple classification problem because artificial neural networks have been widely used for classification problems. They consist of several interconnected layers, which pass information from one layer to another with some modifications.

A deep neural network has an input layer, output layer, decision layer, and hidden layer (one or more). The input layer receives input data in the form of a table. While the neural network can work with raw data, a considerable improvement can be achieved through initial data processing. Normalization of a dataset is applied to scale the numeric values to a common range without distorting the differences in the data. This helps with performance of the machine learning model and also improves its stability. There are various normalization techniques, such as scaling to a range, log scaling, clipping, and more. In this paper, we apply normalization based on the mean and standard deviation of the data Eq. (6).

$$X_n(i) = \frac{X(i) - \mu}{\sigma} \tag{6}$$

Where $X(i)$ and $X_n(i)$ are raw and normalized data. Calculations are applied to each row of the dataset. μ and σ are data mean and standard deviation, calculated over the entire dataset.

Normalized data is then passed to the input layer, which consists of the same number of nodes as the number of features in the dataset.

Data from the input layer is then passed to the hidden layer and transformed into different sets of higher features through a non-linear function. Eq. (7) shows Rectified Linear Unit function (ReLU) which is commonly used for classification problems because of the ease of training and greater accuracy it provides, and will be used in this work (Agarap, 2018):

$$ReLU(x) = \max(0,\ x) \tag{7}$$

Hidden layer transformations are a set of matrix calculations that involve weights, data, and biases, with added ReLU function. A dropout layer is added after each hidden layer for regularization of the neural network to prevent overfitting of the model (Srivastava et al., 2014).

The output of each hidden layer is calculated as shown in Eq. (8):

$$H_O = w_i H_i + b_i \tag{8}$$

where H_O is the output of a hidden layer, H_i is the input of that layer, w_i and b_i are weight and bias vectors. The output of one hidden layer is passed as input to the following hidden layer.

The output of the last hidden layer is collected into an Output Layer that has the same number of nodes as required classes. These values are then transformed into probabilities through a decision layer Eq. (9), such as softmax function (Goodfellow et al., 2016):

$$P_c = \frac{e^y}{\sum e^y} \tag{9}$$

where y is a vector coming from the output layer of the DNN and P_c is the probability assigned to each category c. In the case of fault detection, there are only 2 categories: faulty and faultless operation. The output layer classifies the input data into a category based on the highest probability from P_c. We define accuracy as the number of correctly classified data points Eq. (10):

$$Acc = \frac{\#\ of\ classified\ labels}{\#\ of\ true\ labels} \times 100\% \tag{10}$$

In some applications, we are concerned with the number of false negatives and false positives. In this work, we will use both the accuracy calculation stated above and a confusion matrix that shows the number of actual and predicted samples for each category in one figure (Ting, 2017).

The process of training the network includes the calculation of the weight and bias matrices. Initially, when all values are taken at random, the performance of the network is intolerable. We improve the model through the backpropagation technique, which involves recalculation of weights and biases based on the difference between predicted and actual output. This difference is called cost function, and the backpropagation method tries to minimize it.

Besides choosing the type of backpropagation, we also need to select an appropriate hidden layer structure. We do this by selecting different combinations of several layers. In this work, 2 and 3 hidden layers with various numbers of nodes were tested. The layer structure needs to be complex enough to capture the nonlinear patterns within the data and have greater capacity to prevent

underfitting; but simultaneously, a structure that is too complex may lead to longer training times and an overfitting that prevents it from generalizing to a test dataset.

Both static and dynamic data can be used in DNN. In case of dynamic data types, a "SequenceInputLayer" was used to accept a dataset containing temporal data, and long short-term memory hidden layers were used due to their capability of processing sequential data types.

2.4. Step 3. Classifying faulty data into fault types, i.e., fault diagnosis

Fault diagnosis is an extension to fault detection and can also be performed using DNNs. The procedure for fault diagnosis is the same as in fault detection except for the output layer. The output layer in this step consists of nodes depicting fault type. Each fault is treated as a class, and we simply perform a classification task. The performance of this model is primarily measured by the confusion matrix and classification accuracy Eq. (11). The performance of both networks is combined to calculate overall performance:

$$Overall\ Performance = \frac{\#predicted\ faults}{\#actual\ faults} \times \frac{\#predicted\ fault\ type}{\#actual\ fault\ type} \times 100\% \quad (11)$$

The outcome of this step greatly depends on the accuracy of the previous step. This requires careful tuning of both models, which translates into an increased amount of time needed to set up the hybrid model as there are twice more parameters to configure.

2.5. Integrating all models to produce a hybrid model

Combining all steps, we get a hybrid model which can detect and diagnose faults from complex data (Fig. 1). The first step is needed in case the dimensionality of the dataset is too great. The first step can be omitted otherwise. Then, consecutive deep neural networks perform classification tasks to detect faults and then classifying them by types. All steps are implemented in MatLab consequently, and the model is tested with new data.

3. Applications of the hybrid model

In this section, we discuss the application of the hybrid model to classify the MNIST handwritten digits dataset, which is a benchmark for neural network classification problems (LeCun and Cortes, 2010). We provide different network setups and training options and compare their accuracies to determine their performance. Then, we train the hybrid model to detect and diagnose faults in the Tennessee Eastman (TE) process (Rieth et al., 2017). For illustrative purpose, we trained the model to differentiate between 4 fault types and a non-faulty operation.

3.1. MNIST dataset

MNIST dataset consists of images (28 × 28 pixels) of handwritten single digits and is widely used to test the performance of neural networks. An example of these images is shown in Fig. 2.

While MNIST is used for general classification tasks, it can also be applied to specific areas unrelated to digit recognition. At its core, the classification of handwritten digits is an optimization of network parameters to fit nonlinear patterns; therefore, we can use it as a temporary substitute in other applications where it is difficult to obtain data. We use the MNIST dataset to measure the relative performance of different network setups, which can later be used in fault detection and diagnosis tasks. The dataset is evenly distributed between digits and utilizing a part of the dataset will still yield satisfactory results. The primary constraint of our network is data points, which is why we could not use the Tennessee Eastman Process directly as it is difficult to isolate each fault and reduce the amount of data. Out of 60,000, we used 10,000 to 15,000 to reduce training time and 5000 testing data points.

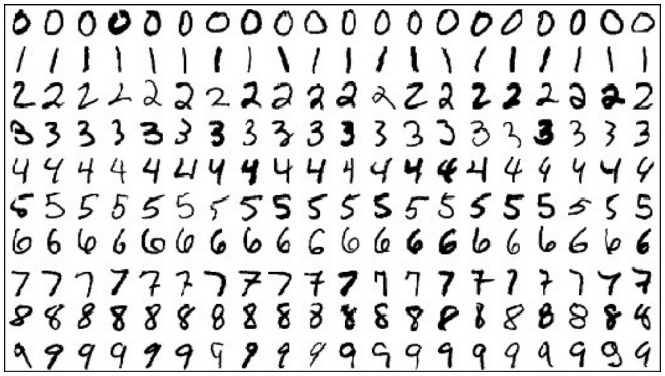

Fig. 2. A selection of images of 10 handwritten digits. Each image is 28 × 28 pixels.

Another advantage of the MNIST dataset is its large dimensionality: the 28 × 28 pixels result in 784 features. The large number of features is usually a downside, but as in the present case, it provides a basis for dimensionality reduction and paves way to show how such datasets can be dealt with through the hybrid model.

3.2. Dimensionality reduction (Step 1)

Having large dimensionality in the input dataset can lead to long training times and reduced accuracy. In some industrial processes, the data may be too complex, and dimensionality reduction may be required. For these reasons, we implement Kernel Principal Component Analysis (kPCA) to reduce dataset complexity. In our model, we use the Radial Basis Function (RBF) kernel to create the kernel matrix (Zhang, 2015). Eq. (12) shows RBF kernel

$$K = \exp\left(-\frac{|x-y|^2}{\gamma}\right) \quad (12)$$

where x and y are two samples representing feature vectors in the input space, and γ is a width parameter. The width parameter was chosen as 10,000, this number was selected after a series of sensitivity analyses aimed to achieve the largest variability contained within the chosen number of principal components. We then select the number of principal components and generate a new *input* dataset. As the original MNIST dataset has 784 variables, we created various datasets ranging from 50 to 500 dimensions to test the effectiveness of the method for various dimensions of the input dataset. There is always some loss of variability when applying PCA techniques; however, reducing dimensionality allows the following neural network to train better.

The main disadvantage of using the kPCA algorithm is its inability to handle data with many data points. Since we have an input matrix X with N data points, the Kernel matrix that we obtain is of B-by-N dimension, which occupies a large portion of computation space (RAM).

3.3. Training networks with different input datasets (Step 2)

This section will use the various datasets created from kPCA to train a network to classify handwritten digits. This will be analogous to the fault diagnosis task as we treat each number as an operational state or a fault state. For proper comparison, we use 12,000 data points for training, 3000 for validation, and 5000 for

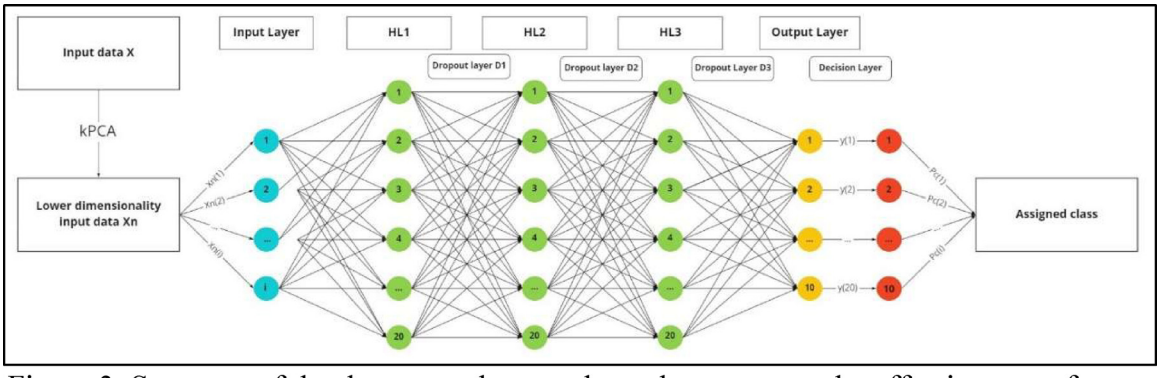

Fig. 3. Structure of the deep neural network used to compare the effectiveness of dimensionality reduction via kPCA method.

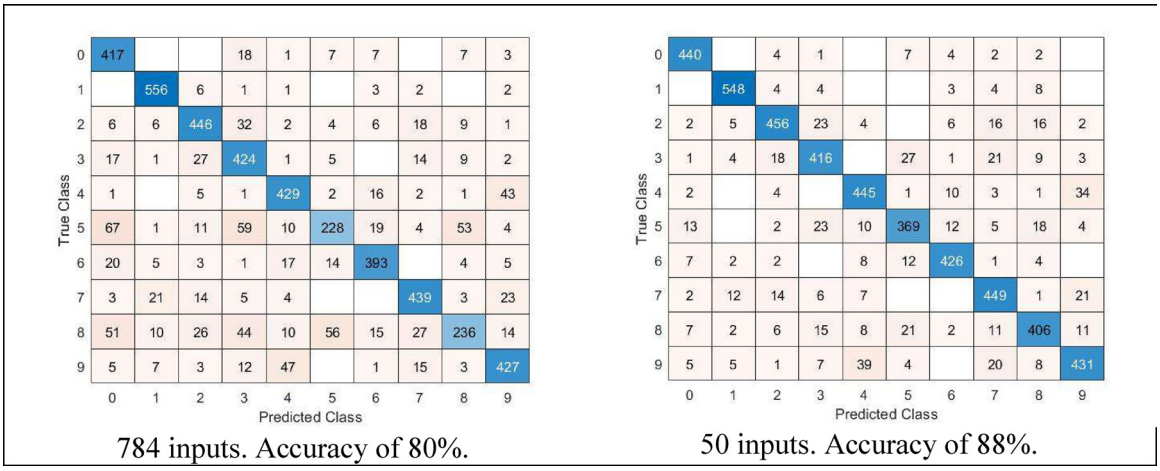

Fig. 4. Confusion matrices for classification of digits with 784 inputs (left) and 50 inputs (right) for MNIST dataset estimation.

testing in all simulations. The original dataset contains approximately 60,000 datapoints, and 15,000 datapoints (25%) were used due to RAM limitations of kPCA algorithm. We assumed that 25% of the dataset is sufficient for producing acceptable results. We utilized ADAM - adaptive moment estimation algorithm (Kingma and Ba, 2015), and networks would train for up to 1000 epochs with occasional premature stops to reduce overfitting. Neural networks consist of three hidden layers with 20 nodes each, a dropout layer after each hidden layer of 20%, and ten output layers depicting ten digits. The input layer would be different each time, depending on the input dataset used. The schematic of our network is shown in Fig. 3. Table 1 shows accuracy results for networks with different input datasets.

The first case without kPCA resulted in 80% classification accuracy. Fig. 4 shows confusion matrices for the initial case and the final case.

Based on Table 1, the accuracy steadily increased as the number of features decreased. Fig. 4 shows considerable improvement for classifying several classes (most notably for numbers 5 and 8). This is considered to be the result of the neural network having less complex input, which allowed for a more accurate tuning of the network parameters.

Table 1
Network structures for different input datasets.

Case #	Number of features	Accuracy
1	784	80%
2	500	80%
3	400	83%
4	300	84%
5	200	85%
6	150	86%
7	100	86%
8	75	86%
9	50	88%

3.4. Dividing fault detection and diagnosis (Step 3)

In some cases, the classification of data into working/fault types may prove difficult. Especially if the dataset is dominated by operational data points, the network may be biased towards the operational class. To overcome this issue, we decided to split the task of detection and diagnosis. To do so, we designed two neural networks in series, as shown in Fig. 5. First, the input data is processed by kPCA algorithm to reduce the number of variables to 50. For this case, we choose 8000 data points for training, 2000 points for validation, and 5000 for testing. This is fed as input data into the first neural network N_I. N_I has only two outputs to differentiate working and faulty conditions. In our case, we labeled digits 0–4 as faulty and digits 5–9 as working. Following N_I, all operational datapoints (digits 5–9) are removed and the second network N_{II} is trained to classify digits 0–4 to simulate fault diagnosis. To compare the results of this hybrid model, we simulated another network that would classify ten digits directly using the same input data and network parameters as in N_I. Table 2 shows the comparison between the two.

From Table 2, the overall performance of the combined model is better than a single model by 2%. Training a single model took roughly 12 min, while each N_I and N_{II} trained for 2–2.5 min.

The results shown represent an incomplete dataset and may contain a bias due to sampling of the data. However, the potential bias would be applied to all model configurations, and could

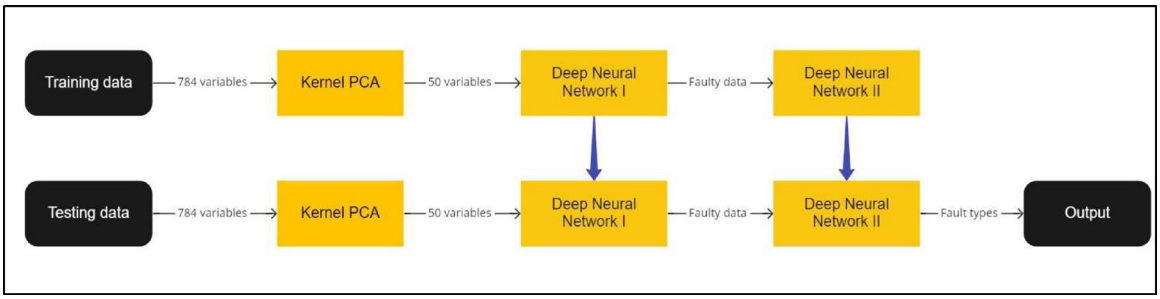

Fig. 5. A detailed flowchart of training and testing data used forfault detection and diagnosis using separate deep neural networks in MNIST digit classification case study.

Table 2
Double network model vs Single network model. *Training time is for comparison purposes. This is dependent on the computer capacity which is training the model.

Model type	*Single model*	*Double model* $N_I + N_{II}$
Input features	50	50
Training data	8000	8000
Validation data	2000	2000
Testing data	5000	5000
Classification accuracy for digits 0-4	84.7%	
Classification accuracy for N1	N/A	91.2%
Classification accuracy for N2	N/A	95.2%
Overall performance	84.7%	86.8%
Training time, minutes*	12	5

Table 3
Selected faults for the case study.

Fault ID	Process variable	Type
Fault #1	A/C feed ratio	Step
Fault #6	A feed loss	Step
Fault #12	Condenser cooling water temperature	Random variation
Fault #18	Unknown	Unknown

be ignored since this is a comparative example that is aimed to explore the effect of the proposed model.

3.5. Fault detection and diagnosis in the Tennessee Eastman (TE) process

The same approach as described above were applied to the TE process to observe the accuracy of fault detection and diagnosis in a chemical process. The datasets published online (Rieth et al., 2017) contain process data of over 500 simulation runs for normal/faulty operation. Original data has 20 faults and 52 input nodes. In this case study, we reduced the amount of data used for training due to RAM requirements, such that we dealt with four fault types (#1, #6, #12, #18) and a non-faulty operation. Table 3 describes the selected faults. In addition, each class was limited to 100 simulations. Overall, this case study is a miniature version of the TE process, aimed to demonstrate the performance of the hybrid model compared to the base model (Heo and Lee, 2018).

The network configuration was kept constant throughout all examples. As such, the hidden layer consisted of three layers, each containing 25 nodes. A dropout layer of 0.2 was added after each hidden layer. The networks were trained until overfitting started, which was generally under 15 or 25 epochs. The network training was manually stopped while monitoring for overfitting, in particular, the data loss progression would go up as the network memorized certain data. The hyperparameters for the network were updated according to batch size, which was selected as 20. This value can affect the accuracy of parameter estimation and was selected after trial and error. Due to reduced number of datapoints, the network training step took several minutes compared to 20 min when using full dataset. However, a significant amount of time (several hours) was required for the data sampling and kPCA dimensionality reduction step.

First, we measured the effectiveness of a single neural network on fault detection and diagnosis (FDD) performance. Applying the single neural network to the reduced TE dataset resulted in a 97.4% accuracy of fault prediction. The model struggled to differentiate between fault #12 and fault #18 in 13 cases out of 500. Due to the reduced dataset, training times for all simulations were shorter.

As a first step, we decided to split the network into fault detection and fault diagnosis. While keeping the same network configurations, we trained the first network N_I to detect and remove all healthy cases; and the faulty cases would then be diagnosed by the second network N_{II}. The first network showed a 100% detection rate, while the second network correctly identified 393 out of 400 faulty cases, leading to a total 98.3% prediction accuracy.

Then, we wanted to investigate the effect of dimensionality reduction on FDD in the TE process. We performed kPCA reduction from 52 to 25 principle components and repeated the network training. The single FDD network was able to accurately predict 497 out of 500 cases, or 99.4%. Applying kPCA reduction to the double model showed a slight improvement as the hybrid model diagnosed 398 out of 400 faults in the second network, with a 100% fault detection accuracy in the first network, bringing the overall accuracy to 99.5%. The results of the simulations are presented in Table 4.

The comparison between confusion matrices of the base model and the enhanced model are shown in Fig. 6.

4. Discussion

Results shown in Table 1 provide a visible effect of reducing input dimensions on the classification accuracy. Since the deep neural network is not the optimal solution for digit classification, we would not expect near-hundred percent accuracy; we also use a limited amount of data for network training, making it even more difficult to classify data points correctly. Because all cases had the same framework, we were able to isolate input dimensionality and compare the results. By reducing the number of variables that the network is learning from, we could increase the initial accuracy of 80% to 86-88%. The 'optimal' number of variables for this case study is between 150 and 50. Reducing dimensionality further may result in unnecessary loss of variability, which can lead to reduced prediction accuracy. This work showed that kPCA can be successfully applied to complex datasets and can result in more accurate classifications. Considering reliability, this can help detect and classify faults with less error, and increasing model accuracy by 10% is a significant upgrade. Applying this technique is suited for industrial process data which is complex.

In this work, we also showed the benefit of splitting the model into fault detection and fault classification. Having the same constraints, we achieved a slight increase in overall classification ac-

Table 4
Results of different model configurations for the TE process.

#	Description	FDD accuracy
1	Single neural network.	97.4%
2	Double neural network configuration.	98.3%
3	Single neural network-assisted by kPCA.	99.4%
4	Double neural network configuration assisted by kPCA (the proposed model)	99.5%

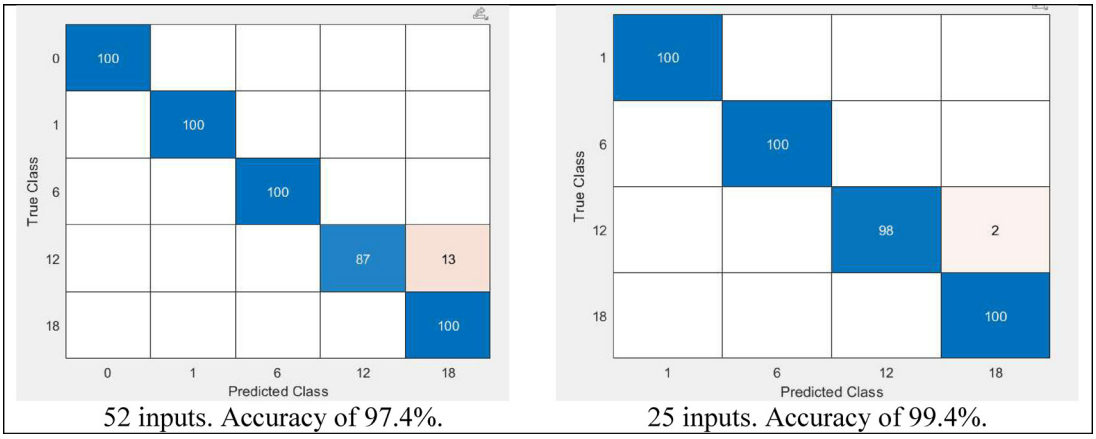

Fig. 6. Confusion matrices for classification of digits with 52 inputs (left) and 25 inputs (right) for TE process estimation.

curacy, which may be beneficial in process data analysis. We initially predicted that the main downside to this model would be the increase in tuning complexity: since we have two models instead of one, we must tune double the parameters. However, by splitting the task into detection and classification, we reduced the number of output nodes, which increased the speed of network training. This allowed quickly to tune parameters for one model, get relevant results, and then focus on the second model. In the end, comparing to a single neural network, we spent far less time working with two networks. Larger datasets may result in longer training times, and therefore will take longer to adjust parameters for optimal results. Providing an option to significantly reduce this time while not losing accuracy is another outcome of this paper.

Then, we applied different model setups for a reduced dataset of the TE process. It was shown that splitting diagnosis and detection into separate neural networks can be successfully applied to a chemical process. It results in a slight increase in classification accuracy, reduced training times, and eases the tuning process. In addition, the application of kPCA as a dimensionality reduction technique to the data helped increase the accuracy even further. By reducing the complexity of the training dataset to 25 variables, the model was able to diagnose faults with 99.4-99.5% accuracy. A similar study achieved 97.3% accuracy using DNN in detection and diagnosis of faults in TEP (Heo and Lee, 2018), however they used a complete dataset. This is similar to our result using base model (97.4%). Although we cannot make direct comparison between studies with complete and incomplete dataset, the improvement from a hybrid approach cannot be overlooked. We conclude that this model could be applied to chemical processes with large dimensionality and produce more accurate results. DNN was used as an example of a ML method in this work, and it is suggested that other combinations of kPCA and ML mechanisms are explored for potential increase in FDD performance.

A downside to this model is that it is not clear how much variability is lost during kPCA. It is suggested to add a step to confirm that the reduced dataset is an accurate representation of the process. Another downside to using kPCA is its difficulty in handling large and/or dynamic datasets. The basic kPCA algorithm is not flexible and does not allow the addition of data to improve the model. For this reason, in the case study, we had to use a simplified version of the TE process. In reality, it is essential that the model is regularly updated and new data is tested in real-time. This can be overcome by utilizing a modification of kPCA, such as two-phase incremental kPCA (TP-IKPCA) (Zhao et al., 2019). The authors concluded that the enhanced model produces similar results to conventional kPCA, but is computationally faster, and it can be used for large and dynamic datasets. Another challenge of the proposed model is that both kPCA and DNN are trial and error dependent. The success of the model will largely depend on the model setup and parameter selection. It is suggested to look into the corresponding literature for kPCA and DNN to determine appropriate model configurations.

5. Conclusion

This paper proposes a hybrid model that combines kPCA and deep neural networks to help detect and diagnose faults. The novelty of the work lies in the successful application of the proposed hybrid model by reduction of the dimensionality of a complex chemical process dataset and splitting the detection and diagnosis task between two DNNs to achieve higher accuracy of data interpretation.

We treated fault detection and diagnosis as classification problems in the MNIST digit classification dataset; and then applied the model to the TE process. The first part of this work concentrated on the application of kPCA to process data, and we showed that reducing the dimensionality of input data had allowed neural networks to train more proficiently, resulting in improved classification accuracy (by approximately 10%). Then, we demonstrated that dividing fault detection and diagnosis into two separate neural networks could further improve overall accuracy and reduce network training times. Using the hybrid model brought the FDD accuracy in the TE process to 99.4-99.5%.

However, while training times were noticeably reduced in the later stages of the hybrid model, the model needs a significant amount of time in the early stage, where input data is processed via kPCA. In addition, the model in its current state cannot work with large amounts of data points and dynamic data. This can be

overcome by adopting variations of kPCA explicitly aimed to cover the weaknesses of conventional kPCA. In this work, we did not focus on the effects of layer configuration and interactions between kPCA and other types of networks, which could be the focus of future research.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Alibek Kopbayev: Conceptualization, Methodology, Software, Investigation, Writing – original draft, Writing – review & editing. **Faisal Khan:** Conceptualization, Methodology, Formal analysis, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Ming Yang:** Methodology, Validation, Formal analysis, Project administration, Writing – review & editing, Supervision, Funding acquisition. **S. Zohra Halim:** Methodology, Formal analysis, Writing – review & editing, Supervision.

Acknowledgments

The authors thankfully acknowledge the financial support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Canada Research Chair (Tier I) Program in Offshore Safety and Risk Engineering.

References

Agarap, A.F. (2018). Deep learning using rectified linear units. CoRR.
Amoozgar, M.H., Chamseddine, A., Zhang, Y., 2013. Experimental test of a two-stage kalman filter for actuator fault detection and diagnosis of an unmanned quadrotor helicopter. J. Intell. Robot. Syst. 70 (1–4), 107–117. doi:10.1007/s10846-012-9757-7, Theory and Applications.
Antonio, J., Geymayr, B., Francisco, N., Ebecken, F., 1995. Fault-tree analysis: a knowledge-engineering approach. IEEE Trans. Reliab. 44 (1), 37–45.
Cimpoesu, E.M., Ciubotaru, B.D., Stefanoiu, D., 2013. Fault detection and diagnosis using parameter estimation with recursive least squares. In: Proceedings of the 19th International Conference on Control Systems and Computer Science. CSCS, pp. 18–23. doi:10.1109/CSCS.2013.35 2013.
Dong, Y., Joe Qin, S., 2018. A novel dynamic PCA algorithm for dynamic data modeling and process monitoring. J. Process Control 67, 1–11.
Gao, D., Zhang, B., Ma, X., Wu, C., 2010. Application of signed directed graph based fault diagnosis of atmospheric distillation unit. In: Proceedings of the 2nd International Workshop on Intelligent Systems and Applications. ISA, pp. 0–4. doi:10.1109/IWISA.2010.5473271 2010.
Ge, Z., Song, Z., Gao, F., 2013. Review of recent research on data-based process monitoring. Ind. Eng. Chem. Res. 52 (10), 3543–3562.
Goodfellow, I., Bengio, Y., Courville, A., 2016. Softmax Units for Multinoulli Output Distributions. In: Deep Learning. MIT Press, pp. 180–184 http://www.deeplearningbook.org.
Ham, D.H., Yoon, W.C., 2001. The effects of presenting functionally abstracted information in fault diagnosis tasks. Reliab. Eng. Syst. Saf. 73 (2), 103–119. doi:10.1016/S0951-8320(01)00053-9.
Heo, S., Lee, J.H., 2018. Fault detection and classification using artificial neural networks. IFAC-Pap doi:10.1016/j.ifacol.2018.09.380, https://doi-org.qe2a-proxy.mun.ca/.
Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., 2012. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Proc. Mag. 29 (6), 82–97.
Huang, S., Tan, K.K., Lee, T.H., 2012. Fault diagnosis and fault-tolerant control in linear drives using the Kalman filter. IEEE Trans. Ind. Electron. 59 (11), 4285–4292. doi:10.1109/TIE.2012.2185011.
Ince, T., Kiranyaz, S., Eren, L., Askar, M., Gabbouj, M., 2016. Real-time motor fault detection by 1-D convolutional neural networks. IEEE Trans. Ind. Electron. 63 (11), 7067–7075. doi:10.1109/TIE.2016.2582729.
Isermann, R., 1993. Fault diagnosis of machines via parameter estimation and knowledge processing—tutorial paper. Automatica 29 (4), 815–835.
Kano, M., Tanaka, S., Hasebe, S., Hashimoto, I., Ohno, H., 2003. Monitoring independent components for fault detection. AIChE J. 49 (4), 969–976.
Kayumov, Z., Tumakov, D., Mosin, S., 2020. Hierarchical convolutional neural network for handwritten digits recognition. Procedia Comput. Sci. 171 (2019), 1927–1934. doi:10.1016/j.procs.2020.04.206.
Kingma, D.P., Ba, J.L., 2015. Adam: a method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, pp. 1–15.
Ku, W., Storer, R.H., Georgakis, C., 1995. Disturbance detection and isolation by dynamic principal component analysis. Chemom. Intell. Lab. Syst. 30 (1), 179–196. doi:10.1016/0169-7439(95)00076-3.
LeCun, Y., Cortes, C. (2010). The MNIST databse of handwritten digits. URL: http://yann.lecun.com/exdb/mnist/. Last access date: November 30, 2021.
Lind, M., 1999. Making sense of the abstraction hierarchy. cognition. Technol. Work 5 (2), 67–81.
Maki, Y., Loparo, K., 1997. A neural-network approach to fault detection and diagnosis in industrial processes. Control Syst. Technol. IEEE Trans. 5, 529–541. doi:10.1109/87.641399.
Nan, C., Khan, F., Iqbal, M.T., 2008. Real-time fault diagnosis using knowledge-based expert system. Process Saf. Environ. Prot. 86 (1 B), 55–71. doi:10.1016/j.psep.2007.10.014.
Navi, M., Meskin, N., Davoodi, M., 2018. Sensor fault detection and isolation of an industrial gas turbine using partial adaptive KPCA. J. Process Control 64, 37–48. doi:10.1016/j.jprocont.2018.02.002.
Rieth, C.A., Amsel, B.D., Tran, R., Cook, M.B., 2017. Additional Tennessee Eastman Process Simulation Data for Anomaly Detection Evaluation, V1. Harvard Dataverse doi/ doi:10.7910/DVN/6C3JR1.
Sarbayev, M., Yang, M., Wang, H., 2019. Risk Assessment of process systems by mapping fault tree into artificial neural network. J. Loss Prev. Process Ind. 60, 203–212. doi:10.1016/j.jlp.2019.05.006.
Schölkopf, B., Smola, A., Müller, K.B., 1998. Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. 10 (5), 1299–1319.
Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, pp. 1–14.
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15 (1), 1929–1958.
Su, Y., Tao, F., Jin, J., Wang, T., Wang, Q., Wang, L., 2020. Failure prognosis of complex equipment with multistream deep recurrent neural network. J. Comput. Inf. Sci. Eng. 20 (2), 1–10. doi:10.1115/1.4045445.
Tang, L., Zhang, S., Yang, X., Hu, S., 2019. Research on prognosis for engines by LSTM deep learning method. Proceeding of the 2019 Prognostics and System Health Management Conference, PHM-Qingdao doi:10.1109/PHM-Qingdao46334.2019.8942976.
Ting, K.M., Sammut, C., Webb, G.I., 2017. Confusion Matrix. In: Encyclopedia of Machine Learning and Data Mining. Springer US, p. 260. doi:10.1007/978-1-4899-7687-1_50.
Venkatasubramanian, V., Rengaswamy, R., Yin, K., Kavuri, S.N., 2003a. A review of fault detection and diagnosis. part III: process history based methods. Comput. Chem. Eng. 27, 327–346.
Venkatasubramanian, V., Rengaswamy, R., Yin, K., Kavuri, S.N., 2003b. A review of process fault detection and diagnosis: part I: quantitative model-based methods. Comput. Chem. Eng. 27 (3), 327–346. doi:10.1016/s0098-1354(02)00162-x.
Wen, L., Li, X., Gao, L., Zhang, Y., 2018. A new convolutional neural network-based data-driven fault diagnosis method. IEEE Trans. Ind. Electron. 65 (7), 5990–5998. doi:10.1109/TIE.2017.2774777.
Wu, X., Liu, Y., 2017. Leakage detection for hydraulic IGV system in gas turbine compressor with recursive ridge regression estimation. J. Mech. Sci. Technol. 31 (10), 4551–4556. doi:10.1007/s12206-017-0901-y.
Yin, S., Ding, S.X., Zhang, P., Hagahni, A., Naik, A., 2011. Study on modifications of PLS approach for process monitoring. In: Proceedings of the IFAC Volumes (IFAC-PapersOnline), 44. IFAC doi:10.3182/20110828-6-IT-1002.02876, PART 1.
Zhang, J., 2015. A complete list of kernels used in support vector machines. Biochem. Pharmacol. Open Access 4 (5).
Zhang, Z., Zhao, J., 2017. A deep belief network based fault diagnosis model for complex chemical processes. Comput. Chem. Eng. 107, 395–407. doi:10.1016/j.compchemeng.2017.02.041.
Zhao, F., Rekik, I., Lee, S.W., Liu, J., Zhang, J., Shen, D., Garcia-Rodriguez, J., 2019. Two-phase incremental kernel PCA for learning massive or online datasets. Complexity doi:10.1155/2019/5937274, 2019.