# FAULT DETECTION AND DIAGNOSIS USING HYBRID ARTIFICIAL NEURAL NETWORK BASED METHOD

by

© Kopbayev Alibek

A thesis is submitted to the

School of Graduate Studies

in partial fulfillment of the requirements for the degree of

**Master of Engineering**

**Faculty of Engineering and Applied Science**

Memorial University of Newfoundland

February 2022

St. Johns, Newfoundland and Labrador

**Abstract**

This thesis proposes a novel approach to fault detection and diagnosis (FDD) that is focused on artificial neural network (ANN). Unlike traditional methods for FDD, neural networks can take advantage of large amounts of complex process data and extract core features to help detect and diagnose faults. In the first part of this work, a hybrid model was developed to improve efficiency and feasibility of neural networks by combining Kernel Principal Analysis (kPCA) and deep neural network. The hybrid model was successfully validated by Tennessee Eastman Process. The second part of the research focuses on a specific application to gas leak detection and classification. In this scenario, a convolutional network (ConvNet) was used as a feature extraction tool prior to network training due to the visual nature of data. The model was shown to accurately predict leaks and leak sizes; furthermore, further model optimizations were performed and evaluated. The proposed approach is superior to other FDD approaches due to its performance and optimization flexibility.

**Acknowledgements**

At first, I would like to express profound gratitude to my supervisors, Dr. Faisal Khan, and Dr. Ming Yang. I got the foremost help from them in any academic or life related situation. Their motivational feedbacks, research ideas, and patience were the key to reach my goal. I would also like to thank my beloved parents, sister, and girlfriend for their continuous support throughout my academic journey. I was away from them for a long time. They always tried to energize me continent miles away with their inspirational and encouraging words. It was difficult for me to get adapted in a new condition after reaching St. John's. However, the friendly people of this wonderful city made my life much easier here. In past two years, I have met many excellent colleagues in the Centre for Risk, Integrity and Safety Engineering (C-RISE). I would like to thank all of them. Specially, I want to thank Dr. Ming for introducing me to Memorial University of Newfoundland and being a great mentor since my Undergraduate years. I would also like to thank the Natural Sciences and Engineering Research Council (NSERC) for providing the funding for my program.

## Contents

**List of figures**

## List of tables

# Chapter 1

# Introduction

## 1.1 Background

Chemical industry has seen several major accidents in the past few decades which resulted in severe damage to environment, economy, and human health. However, these accidents are not frequent and are usually the result of a chain of incidents that happen prior (Venkatasubramanian, Rengaswamy, Yin, et al., 2003a). Therefore, the researchers from industrial and academic fields have focused on dealing with minor accidents in a form of fault detection and root cause analysis. These techniques are implemented to already operational plants to detect and diagnose potential faults that can adversely affect the process or cause damage. A faulty state means that the process or a variable is outside its acceptable range (Himmelblau, 1978). For example, abnormal increase in feed flowrate may results in excess generation of heat during reaction and is considered as a process fault. There are several causes that can be related to the increased feed rate, such as malfunctioning pump or a controlled, and they need to be monitored to prevent the accident. The process of monitoring process parameters and equipment to identify deviations from normal operating state and finding the root causes for such deviations is called fault detection and diagnosis (FDD). Identifying root case is essential to developing proper corrective response. It is essential to detect faults early, as it may propagate throughout the process and cause deviation or failure of unrelated systems. Although there has been a lot of developments around FDD in recent decades, faults persist in process industries which causes loss of human life,

environmental damage and property loss (Nimmo, 1995). Severe damage to equipment can result in halts in production, which can have economic impact on other industries. For example, with the shortage of microchips in 2020 and 2021, phone and car production companies are being limited in their production, which can affect the prices for the customer (Moore, 2021). However, this was exacerbated by recent fire accidents in two major chip production facilities in Japan and China (Kelion, 2021; Shaw, 2021). Considering that process safety and accident prevention has been researched thoroughly and many methods for fault detection and diagnosis are available, these faults still occur globally, and more sophisticated approaches are needed.

Fault detection and diagnosis fall under abnormal event management system (AEM). In fact, FDD are the first two steps of AEM, and the third step is to take appropriate corrective actions to revert the process back to a healthy operating state (Isermann, 2006). As processes get bigger and more process variables are involved, it becomes more difficult to maintain safe operating conditions. In this respect, development of sophisticated models is sought after to increase the diagnosis accuracy (Venkatasubramanian, Rengaswamy, Yin, et al., 2003b). These models are also called hybrid models and they incorporate several techniques that cover certain shortcomings of their individual parts. With recent development in machine learning, artificial neural networks have been included in several hybrid models that can take advantage of the availability of large process data. These methods were shown to differentiate between faulty and healthy operating state and identify root causes for faults with a high degree of certainty (Gharahbagheri et al., 2017; Pérez-Pérez et al., 2021; Travis et al., 2020; Wang et al., 2021; H. Yu et al., 2015; Zeng et al., 2016).

## 1.2 Objectives

With rapid development of machine learning techniques, hybrid model for fault detection and diagnosis with artificial neural networks have been barely explored. Considering the large amount of process data is becoming more available, ANN hybrid-based approaches have a potential to greatly diagnostic capacity in industrial setting. This work aims explore this area of research with proposing two hybrid methods that mainly focus on detecting and classifying faults using supervised neural networks. The main objective of this thesis is to develop a hybrid approach to fault detection and diagnosis for chemical industries incorporating elements of machine learning. Achieving high diagnosis accuracy is the central focus of this research. In addition, this work focuses on potential reduction in computational resources and time needed to train and test the model. This is achieved via these sub-objectives:

I) Detect and diagnose faults using available process data
II) Reduce the complexity of data for more efficient and more accurate neural network training process
III) Incorporate graphic data for fault detection and diagnosis
IV) Develop a methodology that requires minimal expert supervision when it comes to tuning of network parameters

While first paper is focused on dimensionality reduction technique for ease and efficiency of network training, the second paper is about combining different neural network techniques to achieve detection for a specific problem of gas leakage.

### 1.3 Software used

The proposed models were developed and tested using MATLAB R2021a, a numeric computing platform. It is available for all students at Memorial University of Newfoundland and can be downloaded from https://www.mathworks.com/mwaccount using a student license available at https://my.mun.ca/student. Codes for data preparation, network training, validation, testing, and results demonstration were written in MATLAB. Generation of visual data for the second part of the thesis was done using OpenFOAM, a free, open source CFD software; can be downloaded from https://openfoam.org/download/. The simulations were performed on Virtual Machine using Ubuntu 18.04 operating system.

### 1.4 Organization

This thesis consists of five chapters and contains two manuscripts. Chapter 1 briefly discusses fault detection and diagnosis and outlines potential consequences. In this chapter, the objectives of the research are stated, and core simulation software are listed. Chapter 2 covers extensive literature review on fault detection and diagnosis. Conventional methods as well as data-driven approaches are analysed. It also covers recent advancements in ANN-based fault detection and diagnosis techniques and identifies research gaps. In Chapter 3, a hybrid model is proposed using kPCA and Deep Neural Networks (DNN) to detect faulty operating state and classify identified faults. Tennessee Eastman Process benchmark was used to validate the model. This work has been published in Computers and Chemical Engineering on November 22, 2021. Chapter 4 focuses on leak detection in a plant setting using graphical data. The proposed model consists of convolutional network and a long short-term memory layer

network to deal with visual representations of input data. This paper was submitted for publication to Process Safety and Environmental Protection. In Chapter 5, the outcomes of this thesis are summarized and recommendations for future work are given.

# Chapter 2

## Literature review

This chapter provides and overview of existing FDD techniques. Advantages and limitations will be discussed for each approach. The chapter will be split into five parts corresponding to major FDD technique groups: Model-based approaches, knowledge-based approaches, data driven approaches, ANN-based approaches, and hybrid approaches.

### 2.1 Model-based approaches

In the area of model-based fault detection and diagnosis, two steps are necessary to achieve proper process monitoring. The first step is to identify the inconsistencies between true and expected behavior of the system. The second step is to apply a certain decision rule to diagnose the said inconsistency. The concept of inconsistencies is directly tied to redundancy. Hardware redundancy is achieved via redundant sensors and is typically too expensive to implement large-scale. Analytical redundancy, on the other hand, investigates dependencies between process variables and sets algebraic or temporal relationship between system states, inputs, and outputs. This is accomplished using residuals, which are obtained by comparing the actual output of a specific sensor to a calculated value from the mathematical model. The difference between these values is then analyzed to classify the state of the process as either faulty or faultless (P. M. Frank & Ding, 1997). Ideally, absence of redundancy is desired, which would indicate that the system is operating according to the proposed model. Therefore, it

fundamentally requires an explicit mathematical model, that can be either derived from the first principles or based on empirical data. Constructing a model based on first principle may prove to be difficult, as simulating a complex process requires solving multiple equations for conservation of mass, energy, and momentum. Even with advancements in computer processing units, complex systems may be too computationally expensive and time consuming. Taking into consideration that chemical processes are often nonlinear, the complexity is further increased, which makes this method unfeasible. Therefore, empirical methods are often used instead. These methods rely on obtained knowledge and statistics about the process and largely remove connections to physical variables, which makes the diagnosis procedure less intuitive. Residuals obtained from either methods can then be used to identify a faulty operating state (Chow & Willsky, 1984). However, this is not always true due to presence of noise and various uncertainties during modeling procedure. The most popular methods for quantitative model-based approach are diagnostic observers, Kalman filters, parity relations, frequency response models and others (Venkatasubramanian, Rengaswamy, Yin, et al., 2003a, 2003b).

Diagnostic observers method relies on observers that focus on monitoring a specific output of a system. Observer generates a residual that is then evaluated for faults. In normal operation conditions the residuals are also influenced by noise (Severson et al., 2016). Attempts to differentiate between unknown disturbances from faulty state were successful by employing a bank of observers (P. M. Frank & Ding, 1997). In this method, several observers placed for a specific type of fault while ignoring other faults and unknown inputs. For a healthy state, all observers will return the values for

residuals close to zero. Non-zero value will be attributed to noise and therefore ignored. In case of a fault, most observers continue to display near-zero residuals, while a specific observer will have a drastic change in residual. It is then trivial to isolate the fault and continue with diagnosis. To further reduce the effect of noise, filters were designed, such as unknown input observer (UIO) (Frank & Wünnenberg, 1989). Diagnostic observers and UIOs were successfully applied in many processes for FDD (P. Frank, 1994; Kamatchi Kannan et al., 2021; Sotomayor & Odloak, 2005).

Kalman filter (KF) is widely used in chemical industry to estimate process states(Chang & Chen, 1995). It can perform well having Gaussian distribution disturbance. KF, however, does not produce good results when it comes to nonlinear processes, which led to development of modified KF models, such as extended KF and unscented KF (LaViola, 2003; Wan & Vam der Merwe, 2000).

Parameter estimation is another method for fault detection developed by (Isermann, 1997). This approach is based on the assumption that process parameters are affected by faults. Various techniques for parameter estimation exist, such as linear least squares, orthogonal least squares, discrete-time models, time derivatives and more (Young, 1981). Estimated parameters are then compared to those obtained from sensor measurements. Large discrepancies are correlated to faults. A proper threshold needs to be set according to existing noise during normal operating state. This method is effective for small scale system, where parameter dependencies are clear. In the literature, complex models have been developed to tackle this issue (Che Mid & Dua, 2017). However, this may be too computationally expensive and even impossible for large scale industries.

Another popular approach to detect and diagnose faults is parity equation relations (Gertler, 1997). In this method, residual is calculated from the parity (consistency) of the model output with the measured output and inputs. The residual from the parity equation needs to be zero under the assumption of no process uncertainty and ideal mathematical modeling. These, however, are unrealistic expectations of any chemical process, therefore the application of the method is limited (Patton & Chen, 1991). In addition, only linear processes can be analyzed using parity equation relations, which further reduces its applicability. Later works improved the robustness of parity relations FDD and made it a more viable model-based approach (Odendaal & Jones, 2014; Staroswiecki et al., 1993; Zhong et al., 2015).

Other model-based FDD approaches have been used, such as hardware redundancy (Willsky, 1976), input-output relations (Chow & Willsky, 1984), directional (Gertler & Monajemy, 1995) and structural (Gertler & Singer, 1990) residual approaches and more.

While model based FDD methods have several advantages, such as easy and cost-effective implementation for simple linear processes, they have critical limitations. Most of these approaches work under assumption that the process is linear, which makes it impractical for any batch scenario. These models also highly depend on the availability of theoretical basis and require precise model simulations, which can become computationally expensive and time consuming for complex systems. The accuracy of FDD can quickly deteriorate due to modelling errors. In all consideration, application of model-based FDD approaches in large scale chemical plants is not practical (Venkatasubramanian, Rengaswamy, Yin, et al., 2003b).

Developing tools for FDD has been a very active area of research in the past couple of decades. Many tools have been proposed to deal with faults in different ways. This led to agglomeration of these methods into categories, which include model-based methods, knowledge-based methods, data-based methods, and hybrid methods. In addition, with development of machine learning, ANN-based approaches have become a hot topic for research. In this section, knowledge-based and model-based FDD approaches will be reviewed first. Then, data-driven and hybrid methods will be reviewed.

## 2.2 Knowledge based approaches

Knowledge based approaches heavily rely on prior knowledge about a certain process or a system to detect and diagnose abnormal behavior. A cause-and-effect connections are established based on known physics, which are then implemented in a computational software along with logic systems to perform FDD (Venkatasubramanian, Rengaswamy, & Kavuri, 2003). Many knowledge-based systems have been developed, such as rule-based expert system, fault tree analysis (FTA), Bayesian Network (BN), signed graph (SDG) and many more.

Rule based expert systems were pioneers in knowledge-based systems in FDD. They are mostly represented by a code that contained a bunch of if-else rules that were derived prior based on the fundamental knowledge of the process. The computer program is trying to repeat the mental processing of a human to solve a problem. However, this is a rigid system and can only work with information that it was “taught”, therefore any deviations will turn the model practically useless. Nevertheless, this

model can identify a faulty state, track the root cause using the process data and suggest a corrective measure (Chen & Modarres, 1992; Nan et al., 2008). While it is easy to set up and execute, for complex processes the what-if code will grow massively, which can lead to confusion issues. The major downside of this approach is its inability to incorporate physical laws in detection process, which makes the model rigid and not easily adaptable to changes, as it must be manually re-entered upon receiving new information.

An extension to rule based expert approach is case based reasoning approach. It essentially tries to overcome the core limitation of rule based approach by "learning" new faults that it encounters (Kolodner, 1992). This approach has been successfully applied in several works with promising results (Ashley, 2003; Grant et al., 1996; H. Zhao et al., 2017). However, it suffers from another downside of rule-based approach which is chunkiness. Developing a base model for a complex system already requires a large domain of expert input, developing a self-learning model with more rules makes it even more difficult and time consuming.

Fault tree analysis (FTA) is a very common tool in risk, safety, and reliability. In FTA, a tree is constructed using a top-down deductive approach, where the top event is failure of a system, and the bottom part shows basic events. The basic events are assumed to be independent of each other, while top event is directly dependent on other variables (Vesely et al., 1981). In this model, events are interconnected via Boolean logical "AND" and "OR" gates. This allows to quantitatively calculate the propagation of event probabilities from bottom up. Diagnosis is then performed using minimal cut sets, which represent the shortest path to failure (Woodward & Pitbaldo, 2010). While fault

trees are widely used in risk analysis, their use in FDD is limited. Complex processes would require large fault trees, which would be nearly impossible to build while assuming independence of all basic events. This renders FTA impractical for FDD purposes.

Another popular knowledge-based approach is signed directed graphs, or signed digraphs (SDG). This method uses graphical form to represent cause-and-effect relationship between process variables. Process variables are represented as nodes, while the relationship between then is represented by directed arcs from “cause” node to “effect” node. Each node takes on a state comparative to the steady state. SDG model can be derived from a mathematical model, a differential equation, or from a process data (Umeda et al., 1980). The model can use cause-and-effect relationship as well as the underlying model to detect and diagnose faults and was first presented by (Iri et al., 1979). It was then refined and improved in later publications (Yang et al., 2010, 2012).

Improved version of SDG has been developed called possible cause effect graph (PCEG). This is a very similar model except for node structure. In SDG, nodes are given a specific state, while in PCEG nodes are given meaningful descriptions about the variables. This greatly improves root cause analysis as it contains proper knowledge of the process system. More information on PCEG can be found in (Wilcox, 1992) and (Wilcox & Himmelblau, 1994).

Bayesian Network (BN) is another popular technique for reliability analysis and FDD that has proven to successfully tackle processes with uncertainty (Musharraf et al., 2013). In prior models, uncertainty has been a common limitation, which usually restricted their practical use in the field. BN is a probabilistic approach that presents

interactions between process variables in the graphical form (Neapolitan, 2004). It is fundamentally built with a certain degree of "belief" which makes the model flexible in cases of limited or absent data. Bayesian method is used to update the model variables, or weights, which can be used for both risk assessment and root cause analysis (Wilson & Huzurbazar, 2007). Since BN is a probabilistic approach, it is best suited to deal with uncertainty in FDD, unlike other methods. In addition, it was shown to be applicable for complex industrial systems, such as Tennessee Eastman Process (Azhdari & Mehranbod, 2010). BN has gained popularity in fault diagnosis procedures and works have been published with successful implementation of BN (Gonzalez et al., 2015; Huang, 2008; Khakzad et al., 2013). In addition, many hybrid models incorporating BN have also been proposed for root cause analysis. However, BN is static in nature which excludes its applications to dynamic processes. Extensions of BN have been proposed to deal with dynamic processes. In complex systems, calculation of probabilities and interrelations between variables may be too expensive or even impossible. In addition, BN is designed to work with acyclic relations, whereas many variables in a process may be of cyclic nature. In addition, the prior beliefs used in BN can have severe impact on an entire network. This along with statistical distributions choices for data modelling are challenges to constructing a proper network for FDD.

### 2.3 Data driven approaches

Data driven approaches, which are also called history-based approaches, are reliant on available large amounts of process data to derive dependencies that result in faulty operations. The data is typically taken acquired from sensors and test samples. This data is then processed and used to construct or train monitoring or fault detection tools.

Data driven methods are generally divided into qualitative and quantitative based on the type of collected data. Qualitative data-based approaches mostly rely on expert analysis and are generally limited in FDD capacity. Quantitative methods use statistical tools to analyze and extract useful features from the data. In this section, some of the more popular techniques for quantitative data driven approaches will be discussed, such as principal component analysis (PCA), support vector machine (SVM), artificial neural network (ANN), and partial least squares (PLS) (Venkatasubramanian, Rengaswamy, Yin, et al., 2003b). Since these methods mostly rely on vector operations between matrices that involve large datasets, they received high popularity as the computational capacity of computers increased in the last few decades. Performing a statistical analysis on a process with tens of thousands of datapoints and hundreds of variables can just take several hours on a powerful computer workstation.

Among statistical methods, univariate and multivariate methods exist. Univariate methods are more primitive in a way that they directly compare measured process variables to threshold values. As the measured value drifts away from the accepted value, the fault is detected (Mah & Tamhane, 2004). While the simplicity of the univariate method is obvious, it is usually impractical due to its inability to distinguish between operational disturbances and faults with high degree of certainty, which leads to an unacceptable number of false alarms. To counter this, operator supervision is required to maintain detection validity, but may not be optimal for large scale industrial processes (Kourty & MacGregor, 1995).

As the term suggests, multivariate approach is designed to consider multiple variables at the same time. This is achieved through dimensionality reduction technique. The core

of the technique lies in transforming original data into a lower dimensional space without losing valuable contextual information about the process. The effectiveness of dimensionality reduction technique is usually measured in the percent of variability that is retained throughout the transformation (Hu & Yang, 2021). Multivariate analysis can capture the complexity of the process and therefore differentiate basic process disturbances from those caused by faulty operation.

Principal component analysis (PCA) is a dimensionality reduction technique that has been widely used in risk assessment and process monitoring, among other fields (Adedigba et al., 2017; Price et al., 2006; Zadakbar et al., 2012). The method revolves around eigenvalue decomposition or singular value decomposition (SVD) of the original data. This projects the data onto a new space, effectively performing data rotation, which results in new set of variables being orthogonal to each other. The new variables are called Principal Components, or PCs. PCs are numbered according to the respective eigenvalue from highest to lowest (Dunia et al., 1996). The resulting order of PCs represents the relative variance of the original data. For example, if there are 2 principal components with corresponding eigenvalues of 0.7 and 0.3, that means that the first PC accounts for 70% of original data's variability, and the second PC – for 30%. Eigenvectors represent the "rule" by which transformation occurs. For a simple case of 2 PCs, it can be shown geometrically. Figure 2.1 shows a distribution of datapoints for some arbitrary variables; the straight orthogonal lines are drawn to show the transformation that would occur to obtain PC1 and PC2. For higher dimensional data, the core task is to correctly choose the number of PCs as that is the deciding factor to the quality of transformed data. It is important to reduce initial dimensionality by a

large factor and simultaneously retain the variance of the data intact. Another major drawback of PCA is that it is a linear transformation. Graphically, it means that transformation occurs only via straight lines, such as shown in Figure 2.1. On the contrary, if a dataset is presented in a form shown in Figure 2.2, PCA will not yield accurate results. The two most common statistics to detect faults with PCA are $T^2$ and squared prediction error (SPE). Both monitor the deviation from normalcy and can alarm faulty state. However, proper fault diagnosis is difficult to achieve due to "smearing effect" (Joe Qin, 2003). This happens when several PCs are shown to contribute to faults, because they are calculated by matrix multiplication of initial variables. In addition, the ranking of PCs may not always reflect the actual contribution to the fault. These factors cause misdiagnosis and overall reduce the effectiveness of PCA as FDD tool.

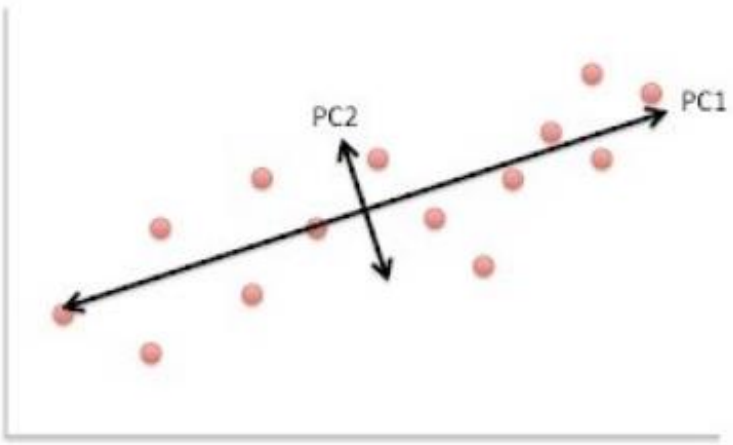



Figure 2.1 Graphical representation of PCA in 2 dimensions (Starmer, 2015)

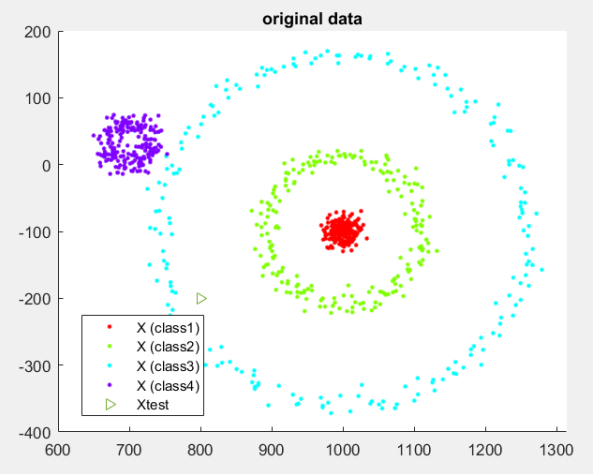



Figure 2.2 Example of a nonlinear dataset that cannot be treated by PCA

In addition, PCA is a static model. Extended versions of PCA were developed to combat nonlinearity, such as kernel PCA (kPCA) (Choi et al., 2005), and static nature of the model, such as dynamic PCA (Ku et al., 1995). Kernel PCA is discussed in detail in Chapter 3.

Support vector machines (SVM) and artificial neural networks (ANN) can be used when sufficient process data is available. ANN is discussed in next Chapter. Data driven FDD tools have shown to be an effective tool for fault detection, but often lack diagnosis precision, especially for low signal faults. More limitations and technology gaps of data-based approaches are presented in Chapter 2.5. To overcome shortcomings of single type FDD methods, hybrid approaches have been developed. Those are discussed in Chapter 2.5.

### 2.4 ANN-based approaches

ANN-based approach is a subset of data-driven approaches, as it highly relies on historical data to perform FDD. It received large popularity in the last decade due to a

lot of development in machine learning (Zhang & Zhao, 2017). ANN-based approaches are divided into supervised and unsupervised learning types, where one requires prior knowledge about fault types, whereas the other does not. Supervised learning has a high FDD efficiency but is harder to set up due to requirements of specific information. Unsupervised models can be applied to a wider range of applications, but their detection and diagnosis accuracy may be lacking. Due to complexity of industrial processes (Ge et al., 2013), ANN-based approaches have been successfully utilized nonlinear process data (Sorsa & Koivo, 1993). A typical neural network has input, output, and one or more hidden layer. Input layer receives data, a hidden layer transforms that data into sets of higher features applying nonlinear function. Output layer converts signals from the hidden layer into probabilities of classes. The connections between layers represent matrix operation with weights and biases, which need to be properly tuned. The tuning of these values is called network training. During network training, the values are initially randomized, and input is fed into the network to calculate the output. The difference between calculated output and true output is compared and the numbers are slightly adjusted accordingly. This process is repeated until the difference between calculated and true result is minimized. Neural networks are generally used for classification purposes and not necessarily for FDD; however, fault detection can be assumed to be a classification problem, setting output classes as “faulty” and “faultless”. The same is applied to diagnosis, as various root causes are simply set as classes of data. This concept was applied to gas leak detection with high accuracy (Wang et al., 2021). While this requires a large amount of specific data, this can also be accomplished using simulated data, as shown in recent research (Travis et al., 2020).

Unlike PCA, neural network can be applied to temporal data using long short-term memory (LSTM) layer networks, which are designed to capture a timed data (D. Song et al., 2021). Neural networks also showed to be compatible with a wide range of data types. (Y. Song & Li, 2021) applied signals in frequency domain to develop several neural network configurations to separate pipeline faults from flow noise. (Ning et al., 2021) used acoustic data to perform FDD in pipelines.

Extensions of neural networks have also been applied to FDD, such as convolutional neural networks (ConvNet), deep neural networks (DNN), recurrent neural networks (RNN) and so on (R. Zhao et al., 2018). (Guo et al., 2016) proposed a hierarchical convolutional network with adaptive learning rate for fault detection and classification. Other variations of ConvNet were successfully used for FDD in mechanical systems (Fuan et al., 2017; Shao et al., 2018).

ANN-based approaches require substantial amount of process data for proper network training. This may not be easily accessible in chemical plants, as it requires a series of sensors for data acquisition, proper transportation, and storage of data. In addition, network training takes a long time due to high computational load, but also needs a great amount of tuning from layer structure to dropout/learning rate/other parameters tuning. In fact, the success of the model will greatly depend on both the quality of data and the tuning process. In general, the tuning process is intuitive as there is no accepted framework. Training for a specific application may take a long time but may also be unattainable due to the model not being able to converge for various reasons. In many cases, data pre-processing can help alleviate these issues, but it adds more complexity to it. Although these issues seem substantial, it is mostly case dependent. Some of these

can be overcome with a specific network which favors certain type of scenario/data type. For example, convolutional networks have stellar performance when it comes to visual data, and LSTM networks can work proficiently with temporal data. Other difficulties have been addressed by combining several FDD approaches to cover various weaknesses of each individual approach.

### 2.5 Hybrid approaches

Hybrid FDD methods are a combination of several independent FDD methods into one. The primary reason for combining FDD methods is that it is practically impossible to construct a diagnostic system that satisfies many requirements of a good system (Mylaraswamy & Venkatasubramanian, 1997). While knowledge-based system excels at diagnostic part, they lack in early detection. History-based approaches are good fault detectors but suffer from misdiagnosis. Model-based systems have high detection accuracy and are very sensitive to fault types; however, they require availability of physical models and are computationally very expensive. Neural networks are generally cheap to set up, but they require substantial information for proper diagnosis and are greatly affected by training parameters. Integrating different models to cover for each others' weaknesses has been more attractive for researchers in recent years (Das et al., 2012). For example, a combination of neural network and expert system was successfully applied to diagnose common faults in a chemical plant (Becraft et al., 1991). Detection of faults was mainly performed by the neural network, while the knowledge expert system identified the root cause and proposed a response plan. Another work suggested to combine neural networks with parity equations (D. Yu et

al., 1996). In this case, neural networks were used to analyze the residuals from parity equations to form a decision upon fault type classification.

A combination of neural network and SDG was proposed to extract the most benefit out of each individual method (Mylaraswamy & Venkatasubramanian, 1997). In this case, neural network was used for timely fault detection, and SDG performed diagnostic role. It performed with high accuracy in the Amoco FCUU process identifying 13 different scenarios. A hybrid FDD method of neural network with extended KF was developed to monitor a chemical reactor (Benkouider et al., 2012). Other works have been conducted to combine neural networks with other FDD approaches, including other neural networks. These hybrid networks consist of several neural network configurations in series, each performing a specific task. For example, (D. Song et al., 2021) proposed a hybrid model that consisted of convolutional network and a LSTM network. Convolutional network was used as a feature extraction tool, while LSTM network was trained to detect and classify faults. In another paper, two convolutional networks were used as both feature extractor and classifier (Ning et al., 2021).

Assuming the complexity of modern processes is increasing, and large amount of process data is easier to collect and store due to advancements in computer technologies, ANN-based hybrid models should be getting more attention in FDD. They have been shown to perform exceptionally well in early fault detection as well as dynamic environment. They can capture hidden connections between variables and classify faults in complex scenarios. Several open-source neural networks have been developed by large tech companies, such as Google, which allows for an easy setup for FDD in some situations.

# Chapter 3

## Fault Detection and Diagnosis to Enhance Safety in Digitalized Process System


### Abstract

The increased complexity of digitalized process systems requires advanced tools to detect and diagnose faults early to maintain safe operations. Availability of online data enables the application of a data-driven approach in fault detection and diagnosis. Deep Neural Networks (DNN) can detect faulty behavior and fault types with high accuracy. Increased system complexity offers challenges in network training, the accuracy of fault detection and diagnosis. Limited work has been done to overcome these limitations. 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.

**Keywords:** Deep Neural Networks; kPCA; fault detection and diagnosis; process system safety; Hybrid model

## 3.1 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 & 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 & 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 dynamic system interactions (Ge et al., 2013; Venkatasubramanian et al., 2003a). A recent direction in a data-driven approach 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 & Zisserman, 2015), and others. It was proposed that neural networks have the potential for fault detection and diagnosis (Z. Zhang & 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 & 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 & 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 & 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 Tennessee Eastman Process (TEP) to confirm the effectiveness in detecting and diagnosing various process faults.

The remaining part of this section is organized as follows. Section 3.2 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 3.3. Section 3.4 contains an analysis and discussion of the results of the case studies. Finally, Section 3.5 concludes the work and gives recommendations for future work.

## 3.2 Methodology to Develop the Hybrid Model

### 3.2.1 Process overview

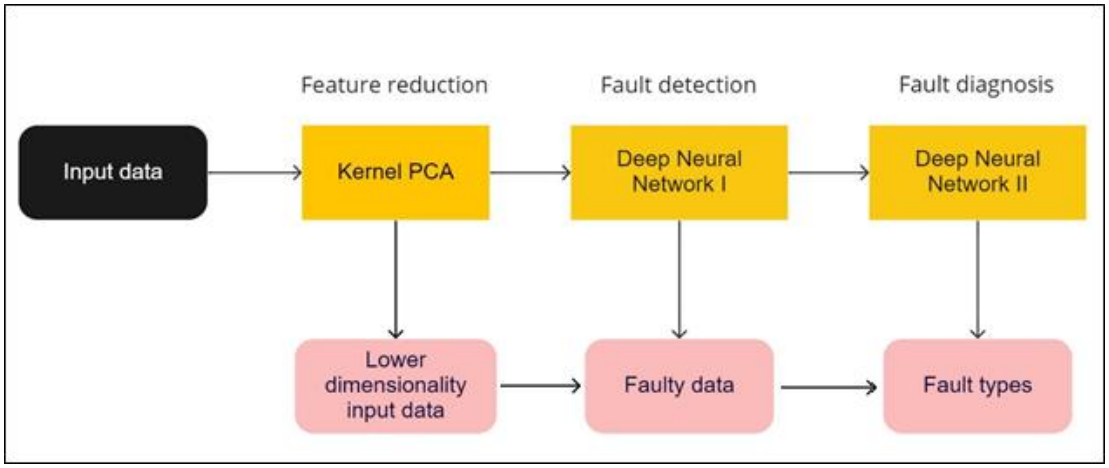



Figure 3.1 Methodology to develop a hybrid model for fault detection and diagnosis using Deep Neural Networks

Figure 3.1 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. For the case study, input data is manually cleaned to remove noise. Effect of noise and randomness on FDD performance of the model needs to be investigated in the future work.

### 3.2.2 Step 1. Reducing the dimensionality of input data with Kernel Principal 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 modelling 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.

First, we map data into a higher feature space ϕ:

$$X(i) \rightarrow \phi(X(i)) \tag{3.1}$$

Then, the covariance matrix becomes:

$$C = \frac{1}{N}\sum_{i=1}^{N} \phi\big(X(i)\big)\,\phi\big(X(i)\big)^T \tag{3.2}$$

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

$$N\lambda \boldsymbol{a} = K\boldsymbol{a} \tag{3.3}$$

Where $N$ is the number of observations, $\lambda$ is eigenvalues, and $\boldsymbol{a}$ is eigenvectors.

However, we must first "centralize" the kernel $K \rightarrow K'$:

$$K' = K - 1_N K - K 1_N + 1_N K 1_N \tag{3.4}$$

where

$$1_N = \frac{1}{N} \tag{3.5}$$

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 (J. 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.

### 3.2.3 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, Deep Neural Network (DNN) is trained 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 as well as improves its stability. While normalization is a common technique used to improve the input data, it carries a risk of losing some features of process parameters. In the case study presented in this chapter, normalization was successfully used by researchers without loss of features (refer to Ch. 3.3.5). 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.

$$X_n(i) = \frac{X(i) - \mu}{\sigma} \tag{3.6}$$

Where $X(i)$ and $X_n(i)$ are raw and normalized data. Calculations are applied to each row of the dataset. $\mu$ and $\sigma$ 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 nonlinear function. Rectified linear unit

(ReLU) 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):

$$ReLU(x) = \max(0, x) \tag{3.7}$$

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 follows:

$$H_O = w_i H_i + b_i \tag{3.8}$$

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, such as softmax function (Goodfellow et al., 2016):

$$P_c = \frac{e^y}{\sum e^y} \tag{3.9}$$

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):

$$Acc = \frac{\#\ of\ classified\ labels}{\#\ of\ true\ labels} * 100\% \tag{3.10}$$

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.

### 3.2.4 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 classification portion of DNN acts as a FDD in this case. 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:

$$Overal\ Performance = \frac{\#\ predicted\ faults}{\#\ actual\ faults} * \frac{\#\ predicted\ fault\ type}{\#\ actual\ fault\ type} * 100\% \tag{3.11}$$

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.

### 3.2.5 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 (Figure 3.1). 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.

### 3.3 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 & Cortes, 2010). Different network setups and training options are examined and comparison between their accuracies was analysed to determine their performance. The hybrid model is then trained to detect and diagnose faults in the Tennessee Eastman (TE) process (Rieth et al., 2017). For illustrative purpose, we the model is trained to differentiate between 4 fault types and a non-faulty operation.

#### 3.3.1 MNIST dataset

MNIST dataset consists of images (28x28 pixels) of handwritten single digits and is widely used to test the performance of neural networks. An example of these images is shown in Figure 3.2.

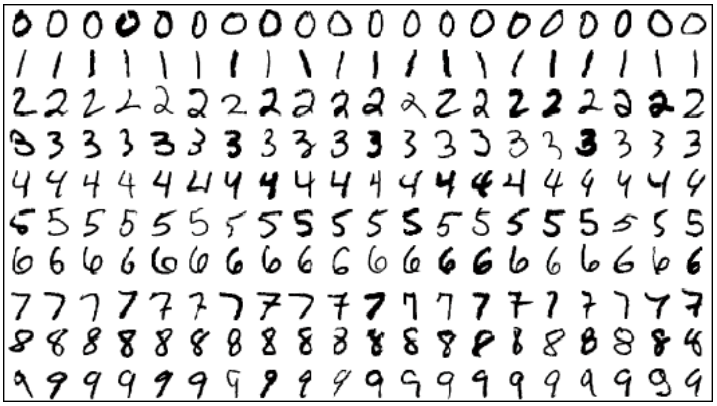


Figure 3.2 A selection of images of 10 handwritten digits. Each image is 28x28 pixels

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, it can be used it as a temporary substitute in other applications where it is difficult to obtain data. The MNIST dataset is used to measure the relative performance of different network setups, which can later be utilized 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 60000, we used 10000 to 15000 to reduce training time and 5000 testing data points.

Another advantage of the MNIST dataset is its large dimensionality: the 28x28 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.

### 3.3.2 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, Kernel Principal Component Analysis (kPCA) is implemented to reduce dataset complexity. In our model, the Radial Basis Function kernel is used to create the kernel matrix (J. Zhang, 2015):

$$K = \exp\left(-\frac{|x-y|^2}{\gamma}\right) \tag{3.12}$$

where *x* and *y* are two samples representing feature vectors in the input space, and $\gamma$ is a width parameter. The width parameter was chosen as 10000, 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, various datasets were created 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).

### 3.3.3 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 12000 data points for training, 3000 for validation, and 5000 for testing in all simulations. As the original dataset contains approximately 60,000 datapoins, we assume that 25% of that dataset is sufficient for producing acceptable results. 15,000

datapoints were used due to RAM limitations of kPCA algorithm. We utilized ADAM - adaptive moment estimation algorithm (Kingma & 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 Figure 3.3. Table 3.1 shows accuracy results for networks with different input datasets.

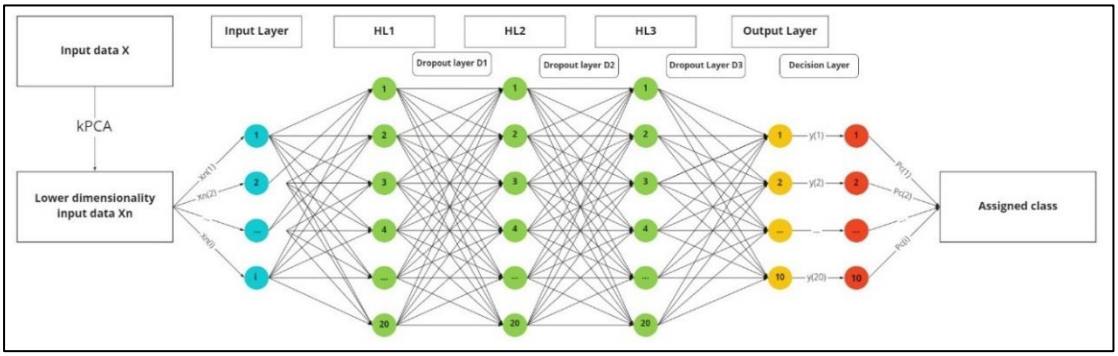



Figure 3.3 Structure of the deep neural network used to compare the effectiveness of dimensionality reduction via kPCA method

Table 3.1 Network structures for different input databases

| 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% |

The first case without kPCA resulted in 80% classification accuracy. Figure 3.4 shows confusion matrices for the initial case and the final case.

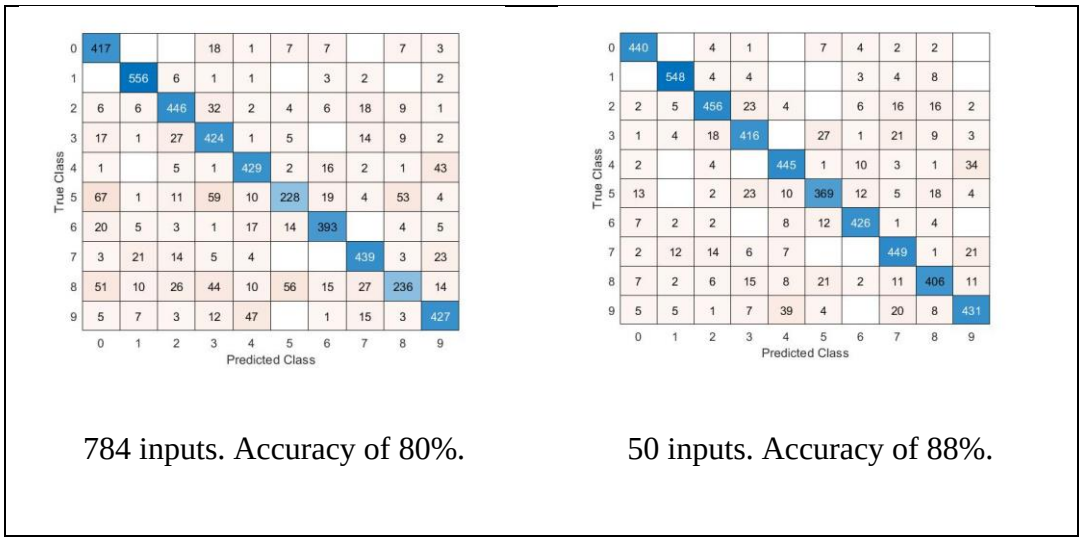



Figure 3.4 Confusion matrices for classification of digits with 784 inputs (left) and 50 inputs (right) for MNIST dataset estimation

Based on Table 3.1, the accuracy steadily increased as the number of features decreased. Figure 3.4 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.

### 3.3.4 Dividing fault detection and diagnosis (Step 3)

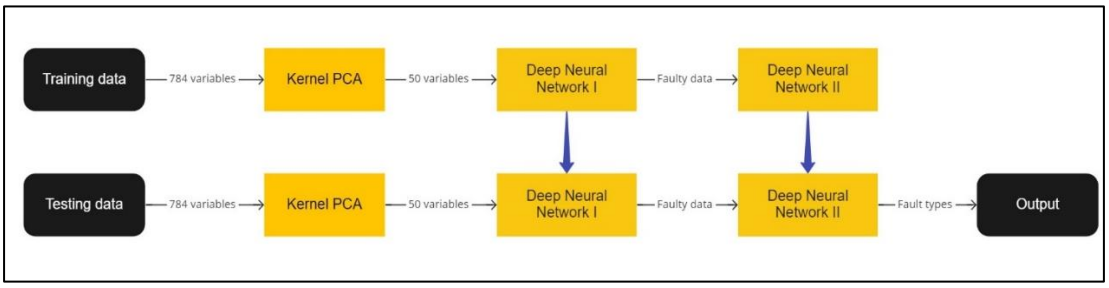



Figure 3.5 Training and testing of fault detection and diagnosis using separate deep neural networks

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, a decision was made to split the task of detection and diagnosis. To do so, two neural networks were put in series, as shown in Figure 3.5. First, the input data is processed by kPCA algorithm to reduce the number of variables to 50. For this case, 8000 data points were chosen 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 3.2 shows the comparison between the two.

Table 3.2 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 |

From Table 3.2, the overall performance of the combined model is better than a single model by 2%. Training a single model took roughly 12 minutes, while each $N_I$ and $N_{II}$ trained for 2-2.5 minutes. 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 be ignored since this is a comparative example that is aimed to explore the effect of the proposed model.

### 3.3.5 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 3.3 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 & Lee, 2018).

Table 3.3 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 |

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 minutes when using full dataset. However, a significant amount of time (several hours) was required for the data sampling and kPCA dimensionality reduction step. It is important to mention that in the context of industrial application of neural networks, the tuning of network parameters needs to be done by hand using trial and error method, which may take a long time or be ineffective. These parameters may need to be revised later to further optimize the model. Absence of a concrete methodology for parameter tuning is a major limitation of such approach.

First, the effectiveness of a single neural network on fault detection and diagnosis (FDD) performance was measured. 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, the first network $N_I$ was trained to

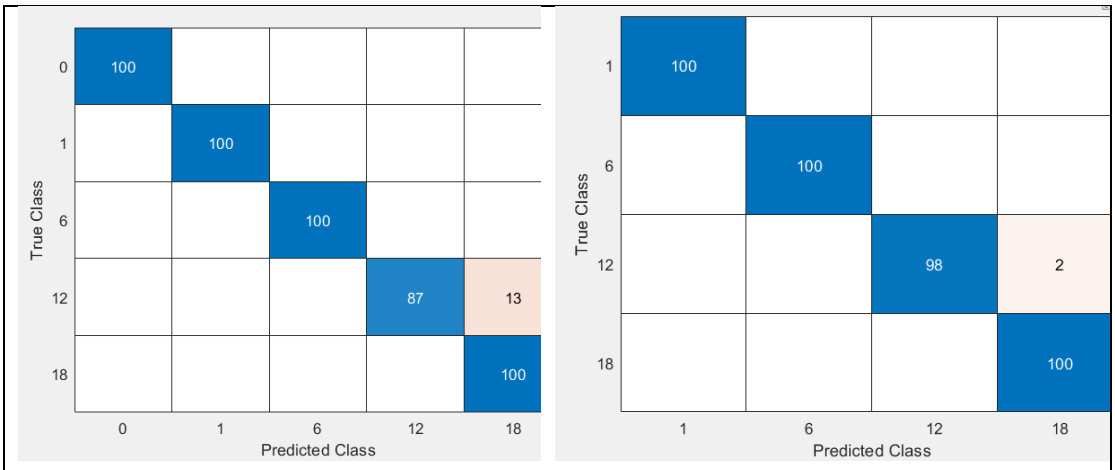
detect and remove all healthy cases; and the faulty cases would then be diagnosed by the second network $N_{II}$. The first network showed a 100% detection rate, while the second network correctly identified 393 out of 400 faulty cases, leading to a total 98.3% prediction accuracy.

Then, the effect of dimensionality reduction on FDD in the TE process was investigated. We performed kPCA reduction from 52 to 25 principal 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 3.4.

Table 3.4 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% |

The comparison between confusion matrices of the base model and the enhanced model are shown in Figure 3.6.




52 inputs. Accuracy of 97.4%. 25 inputs. Accuracy of 99.4%.

Figure 3.6 Confusion matrices for classification of digits with 52 inputs (left) and 25 inputs (right) for TE process estimation.

### 3.4 Discussion

Results shown in Table 3.1 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 accuracy, 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 number of 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 & 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 can 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.

### 3.5 Conclusions

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.

## 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. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *70*(1–4), 107–117. https://doi.org/10.1007/s10846-012-9757-7

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. *Proceedings - 19th International Conference on Control Systems and Computer Science, CSCS 2013*, 18–23. https://doi.org/10.1109/CSCS.2013.35

Dong, Y., & Joe Qin, S. (2018). A novel dynamic PCA algorithm for dynamic data modeling and process monitoring. *Journal of 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. *Proceedings - 2010 2nd International Workshop on Intelligent Systems and Applications, ISA 2010*, 0–4. https://doi.org/10.1109/IWISA.2010.5473271

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* (pp. 180–184). MIT Press. http://www.deeplearningbook.org

Ham, D. H., & Yoon, W. C. (2001). The effects of presenting functionally abstracted information in fault diagnosis tasks. *Reliability Engineering and System Safety*, *73*(2), 103–119. https://doi.org/10.1016/S0951-8320(01)00053-9

Heo, S., & Lee, J. H. (2018). Fault detection and classification using artificial neural networks. *IFAC-Pap*. https://doi-org.qe2a-proxy.mun.ca/10.1016/j.ifacol.2018.09.380

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 Transactions on Industrial Electronics*, *59*(11), 4285–4292. https://doi.org/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 Transactions on Industrial Electronics*, *63*(11), 7067–7075. https://doi.org/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 Computer Science*, *171*(2019), 1927–1934. https://doi.org/10.1016/j.procs.2020.04.206

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.

Ku, W., Storer, R. H., & Georgakis, C. (1995). Disturbance detection and isolation by dynamic principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *30*(1), 179–196. https://doi.org/10.1016/0169-7439(95)00076-3

LeCun, Y., & Cortes, C. (2010). *MNIST handwritten digit database*.

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 Systems Technology, IEEE Transactions On*, *5*, 529–541. https://doi.org/10.1109/87.641399

Nan, C., Khan, F., & Iqbal, M. T. (2008). Real-time fault diagnosis using knowledge-based expert system. *Process Safety and Environmental Protection*, *86*(1 B), 55–71. https://doi.org/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. *Journal of Process Control*, *64*, 37–48. https://doi.org/https://doi.org/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 ed.). Harvard Dataverse. https://doi.org/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. *Journal of Loss Prevention in the Process Industries*, *60*, 203–212. https://doi.org/10.1016/j.jlp.2019.05.006

Schölkopf, B., Smola, A., & Müller, K. B. (1998). Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, *10*(5), 1299–1319.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–14.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *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. *Journal of Computing and Information Science in Engineering*, *20*(2), 1–10. https://doi.org/10.1115/1.4045445

Tang, L., Zhang, S., Yang, X., & Hu, S. (2019). Research on Prognosis for Engines by LSTM Deep Learning Method. *2019 Prognostics and System Health Management Conference, PHM-Qingdao 2019*. https://doi.org/10.1109/PHM-Qingdao46334.2019.8942976

Ting, K. M. (2017). Confusion Matrix. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of Machine Learning and Data Mining* (p. 260). Springer US. https://doi.org/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. *Computers and Chemical Engineering*, *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. *Computers & Chemical Engineering*, *27*(3), 327–346. https://doi.org/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 Transactions on Industrial Electronics*, *65*(7), 5990–5998. https://doi.org/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. *Journal of Mechanical Science and Technology*, *31*(10), 4551–4556. https://doi.org/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 *IFAC Proceedings Volumes (IFAC-PapersOnline)* (Vol. 44, Issue 1 PART 1). IFAC. https://doi.org/10.3182/20110828-6-IT-1002.02876

Zhang, J. (2015). A Complete List of Kernels Used in Support Vector Machines. *Biochemistry & Pharmacology: Open Access*, *4*(5).

Zhang, Z., & Zhao, J. (2017). A deep belief network based fault diagnosis model for complex chemical processes. *Computers and Chemical Engineering*, *107*, 395–407. https://doi.org/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*, *2019*. https://doi.org/10.1155/2019/5937274

# Chapter 4

## Gas Leak Detection Using Combined Neural Network Model


### Abstract

Natural gas leakage has been a major issue in the chemical industry. Methods to detect and diagnose leaks have been developed and widely used for gas pipeline and storage. Most techniques include manual inspection of sensor-aided mathematical models. Although machine learning has seen a lot of developments in recent years, its application in gas leak detection has been barely explored. In this context, artificial neural networks can be used as a feature extraction tool, as a decision-making tool, or both. In this work, we combined a pretrained convolutional network and a long short-term memory layer network to perform both roles, respectively. Our model was trained and tested using sequences of concentration profiles generated by a CFD software. The model learned to successfully predict gas leak and classify its size based on 50 seconds of data. We then looked at flexibility of our network to work with limited amount of data and performed some tuning to have a quick detection model under 30 seconds. The neural networks did not require parameter adjustments to achieve high prediction accuracy, but we showed that further optimization is possible through data selection and preparation. The model needs to be further tested for various leak locations, as this was not covered in this paper due to limitations of CFD software. Experimental results are needed to confirm the effectiveness of the model and use of other types of input needs to be investigated.



**Keywords:** gas leak detection, convolutional neural network, gas leak safety design.

## 4.1 Introduction

Natural gas has become an integral part of society, being widely used for both residential and industrial purposes. Many systems were built to sustain the demand, such as pipeline networks, loading and receiving terminals, storage vessels etc. All these systems are prone to deterioration due to corrosion or aging, which become more prevalent as time goes on. This is associated with one of the primary issues in chemical industry – gas leakage (Eckerman, 2005). Leakage of natural gas can result in severe environmental impact as it is a major greenhouse gas. In addition, it can result in intoxication, suffocation or explosion which results in damage to human health, property, reputation, and finances (Bonvicini et al., 2015). To avoid these consequences, the safety of these systems must be developed and polished. While inspection and maintenance can improve the reliability of pipelines and storage vessels, it is practically impossible to avoid gas leaks. Therefore, an early response plan needs to be put in place to conceal the leak and prevent escalation (Datta & Sarkar, 2016). The first and crucial step is detection of gas leak, which has attracted a lot of attention from industry and research lately.

Several methods are used to detect gas leakage, which rely on manual inspection of pipelines and vessels. These methods require investments, labor and are not continuous in nature. In addition, increasing pipe distance and plant structure complexity, further reduces effectiveness of manual methods. Other methods include monitoring of process parameters, such as pressure, temperature, and flowrate (Xiao et al., 2018). These methods heavily rely on proper data acquisition and the accuracy of the mathematical models (Doshmanziari et al., 2020). Specialized sensors are used for this purpose, such

as acoustic, optical, electrochemical, distributed sensing fiber optic, and so on (Meribout et al., 2020). It was suggested to use image processing methods for gas leak monitoring via infrared cameras that can detect methane molecules on the infrared spectrum (Fahimipirehgalin et al., 2021; Vollmer & Möllmann, 2017).

In recent years with the rapid development of machine learning techniques, neural networks have become a popular in the context of outlier detection. A number of papers have been published in regards to application of neural networks for detection of gas leaks (Ning et al., 2021; Pérez-Pérez et al., 2021; S. J. Song & Jang, 2018; Y. Song & Li, 2021; Travis et al., 2020; Wang et al., 2021). These papers focus mainly on pipeline leak detection and involve various types of neural networks, such as convolutional neural networks, recurrent neural networks, hybrid networks etc. For example, artificial neural network was to enhance the precision of frequency analysis in pipeline leak detection (Wang et al., 2021). The trained model achieved high accuracy but required a series of filters and signal transformation techniques. In another work, artificial neural network was used to predict near real time gas leak at a testing site using both simulated and field data (Travis et al., 2020). However, as in many cases with machine learning, this model was highly dependant on sensor data and interference of unexpected winds caused the model to overestimate the leak rates by a significant value. A separate paper investigated application of artificial neural network in pipe leak detection (Pérez-Pérez et al., 2021). They achieved high accuracy in detection of leaks and leak locations. However, the model required pressurized flow and was highly dependant on network configuration.

A combined network of convolutional layer and convolutional long short-term memory (LSTM) layer was employed predict leak locations in an enclosed space (D. Song et al., 2021). In this work, convolutional layer extracted special representations, while LSTM layer extracted temporal representations. The model was successful in adapting to building layout but did not account for the leak size and used input data not verified by an experiment. The authors did not utilize "healthy" state where no leak occurred, and misclassifications were not explored. In another work, convolutional network was used as a primary detection mechanism for gas leaks in galvanised steel pipe (Y. Song & Li, 2021). The authors investigated many network architectures to see the effectiveness in separating leak detection from internal flow noise in frequency domain. The difficulty of this model lied in the necessity of substantial data denoising and preprocessing, and it required input produced by a specific sensor. Another paper used convolutional network as both feature extractor and classifier in a gas pipeline setting (Ning et al., 2021). The network was aided by spectrum enhancement, which improved prediction accuracy and reduced training times. It also uses sound signals which may not be available for every pipe/vessel in industrial setting and covers a limited number of leak types. Limited work has been done to analyze visual data with neural networks and its application to a leak scenario in a plant setting.

This work proposes a combined model that primarily focuses on leak detection and classification in case of a receiving terminal. It uses a pretrained GoogLeNet convolutional network as a feature extraction tool and a bidirectional long short-term memory layer network (BiLSTM) as a classification tool. Training and testing data were generated using a CFD software that was previously validated by field tests. This

model employs visual input data and tries to optimize the speed of leak detection to reduce escalation and damage. The next part of this paper will explain data generation procedure, the role of neural networks. Classification results will be presented in the following section, and different setups will be evaluated to optimize leak detection speed. Limitations of the proposed model will be discussed and suggestions for future work will be given.

## 4.2 Methodology

### 4.2.1 Process overview

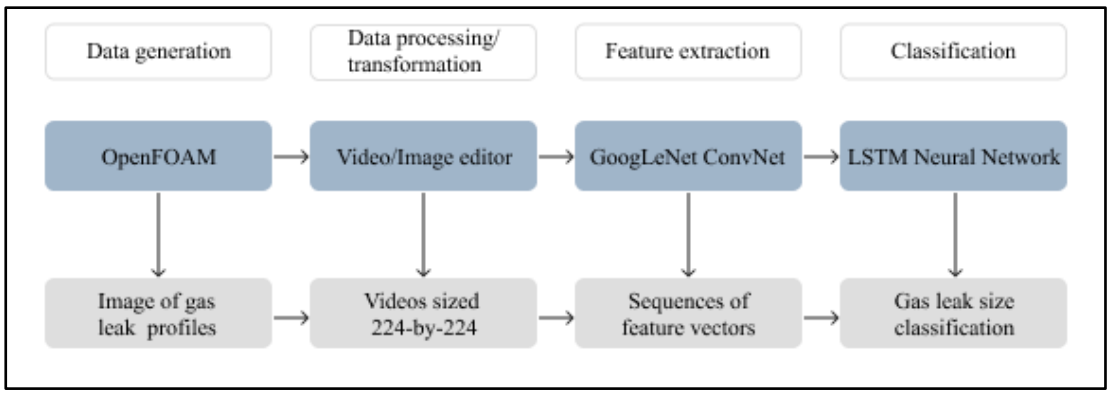



Figure 4.1 Methodology for gas leak detection using a combination of ConvNet and LSTM

Figure 4.1 shows that the process consists of four steps: data generation using OpenFOAM, data processing and transformation, feature extraction using GoogLeNet, and gas leak classification using LSTM neural network. OpenFOAM software is used to generate a gas leak scenario in a receiving terminal and allows to track the changes that happen within first 50 seconds of the accident. It generated a series of images that were then transformed into videoclips of a particular resolution via MATLAB

functions. Video footage is then fed into GoogLeNet convolutional network to extract sequences of feature vectors. These sequences of features are used in bidirectional long short-term memory layer neural network (BiLSTM) training that learns to classify those sequences into gas leak categories. This model utilized strengths of convolutional networks as they are suitable for feature extraction from visual data, which makes network training produce accurate results. Since data generated from OpenFOAM was proven reliable based on previous studies, the combined model should produce promising results.

### 4.2.2 Generation of training data

Prior to building the model, a search for available data was performed on gas leaks that could be used for network training and testing. It appears there is not enough resources freely available online that could satisfy the neural network training requirements. Although Convolutional Network has the possibility to generate additional data for proper training (Jain, 2017), there is a bare minimum that could not be met. Therefore, a mathematical simulation of a gas leak published recently and validated with a twin experiment (Wu et al., 2021) was used in this work. We contacted one of the authors of the paper and received a copy of the model, which was later used to generate sufficient data for network training.

The model was simulated using OpenFOAM, a free open source CFD software, which was installed onto Ubuntu 18.04 operating system inside a virtual machine. The upside is that it is ready for use immediately after installation of the VMware, and some key parameters are easily configured. The downside is that the generation speed is limited

by the virtual machine characteristics and could not be sped up using a more computationally powerful engine. Another downside was inability to change the geometry of the simulation since it was built and validated for that specific geometrical setup. Changing the setup could potentially lead to inaccurate results.

In this work, we will not go over the model in detail as it is extensively described in the original paper. In short, it is a three-dimensional model that combines computational fluid dynamics (CFD) with the ensemble Kalman filter (EnKF) to simulate leaked $CH_4$ vapor propagation in a typical receiving terminal setting. The results of the model were compared to the field test (the Burro 8 spill test) conducted by the Lawrence Livermore National Laboratory (LLNL) at the Naval Weapons Center. In addition, CFD simulations were compared to an ANSYS FLUENT simulation. These tests verified that CFD and EnKF coupling reproduced accurate results and *rhoReactingBuoyantFoam* solver can be used as an alternative to simulate LNG vapor dispersion.

Generation of data was performed using a receiving terminal setup with complex layout, obstacles, buoyancy forces etc. Figure 4.2 taken from the original paper shows the layout of the receiving terminal and the location of the leak (Wu et al., 2021). More details regarding geometrical setup, input parameters and boundary conditions can be found in the paper. To train a neural network, it is required to provide data of different scenarios, such as different leak location or leak size. Since we did not have a chance to change the geometry of the simulation, we decided to set the leakage velocity as the variable parameter. For our simulations, we chose five setups: no leak; 10m/s, 20 m/s, 30 m/s and 50 m/s leakage velocity. Because the leak size can be directly tied to the

severity level, these scenarios reflect different states of the terminal: safe state, unsafe state with variable danger level. All scenarios were simulated within first 50 seconds of the leak, which would allow for quick response to the accident.

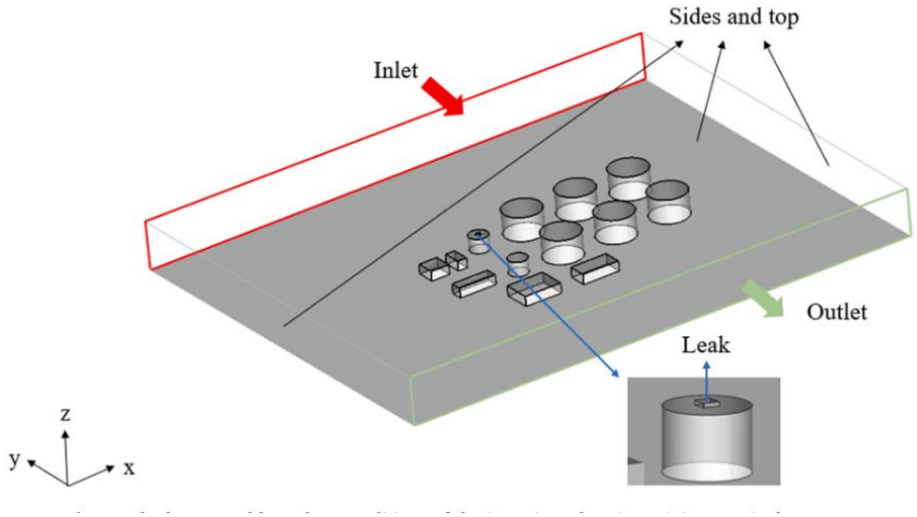



Figure 4.2 The layout of the receiving terminal used for gas leak detection tests (Wu et al., 2021)

The results of each simulation contained 51 PNG files representing concentration profile of a plant at elevation of 20 meters. The 1$^{st}$ PNG is the state of the system at t = 0, where the leak starts. 2-51 files represent the following 50 seconds. Figure 4.3 shows example of the concentration profile for 30 m/s scenario at t = 35 seconds; the axes and color description were removed for clearance purposes. The simulations were repeated 25 times for each scenario; 20 for network training and 5 for network testing. In total, there were 125 simulations containing 6375 PNG files. There was no identical simulation as each time the leakage velocity was increased or decreased by a small amount: for example, simulations for 30 m/s contain simulations from 29 m/s to 31 m/s.

In addition, turbulence-related coefficients were changed accordingly for each simulation.

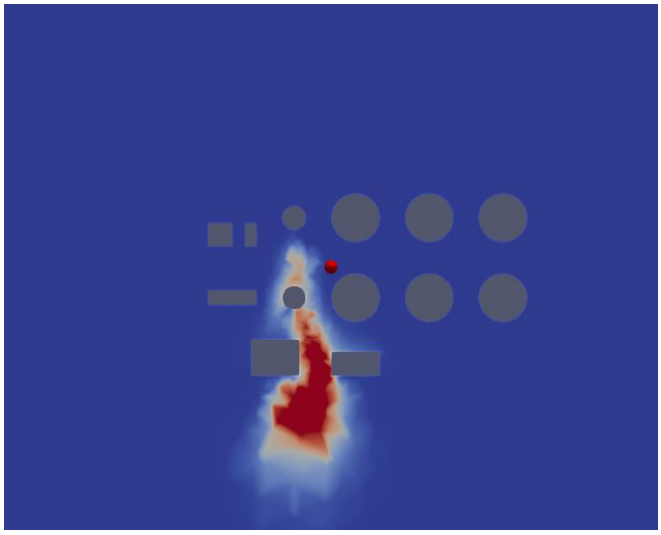


Figure 4.3 LNG vapor dispersion at t = 35s and v = 30 m/s

### 4.2.3 Data Transformation

The generated data consisted of several thousands of images 759 by 610 pixels, which was the default setting for the simulation in OpenFOAM software. This format and amount of data is not convenient to use for network training, as the images are not independent and are sequences of different case scenarios. To simplify the input process, the data was first transformed into video format, hence generating only 100 files for training and 25 for testing.

First, it was important to make sure that all images are homogenous. After each simulation was performed, we made sure that the format and dimensions of the image did not alter. Then, we checked the integrity of all files; some files were corrupt in the process of either results generation or copying from virtual machine to a local disk. In

case some files were damaged, the simulation was repeated, and new results were generated to replace corrupted ones. Finally, sets of images were then transformed into videos using MATLAB *writeVideo* function at 17 frames per second to produce 3 second clips for each simulation.

Finally, the videoclips were resized to fit the input resolution of the GoogLeNet network using additional function that would transform 759 by 610-pixel frames into 224 by 224-pixel frames. Example of the initial and resized frame is shown in Figure 4.4. This process was conducted twice: 100 cases for network training and 25 cases for network testing.

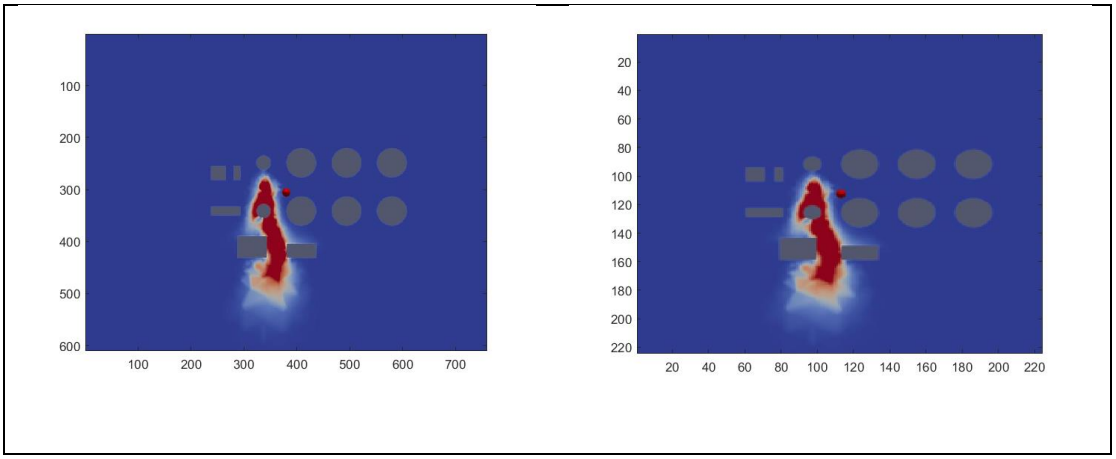


Figure 4.4 Initial and resized versions of the frame at t = 35s and v = 20 m/s

### 4.2.4 GoogLeNet Feature Extraction

Gas leak detection is an integral part of accident mitigation and escalation prevention. In this work, we train a Convolutional Neural Network (ConvNet) to detect gas leaks as well as leak size. We gathered labeled data to assist the training process as an example of supervised learning. Gas detection is treated similarly to a classification problem that artificial neural networks have been widely used for; and since our input

data consists of visual content, ConvNet was chosen accordingly, as it is well suited for such input type.

A typical neural network consists of an input layer, output layer and a hidden layer (one or more). Input data enter the input layer, goes through hidden layer(s), and is then passed to the output layer. Hidden layers transform data into sets of higher feature vectors through nonlinear operations; in case of ConvNets, hidden layers perform convolution operations that generate feature maps that become input of the next layer. In some cases, hidden layers may contain pooling layers, fully connected layers, and normalization layers.

Convolutional network can be used for classification tasks or just feature extraction; and they can be built from scratch or adapted from existing networks. As such, we took advantage of existing GoogLeNet convolutional neural network that is primarily used for image classification developed by researchers at Google (Szegedy et al., 2015). It is a complex network that consists of 27 layers and includes convolution layers, pooling layers, inception modules, dropout layers, output layer and a SoftMax layer. The input layer accepts images of 224 by 224 pixels while the output layer was initially designed to classify 1000 different images, and hence had an output layer of 1000 cells. In our model, we use GoogLeNet as a feature extraction tool; extracted features are then sent to long short-term memory (LSTM) neural network for classification.

To extract features out of frames in each video, we use GoogLeNet's last pooling layer "pool5-7x7_s1" and *activations* function to generate sequences of feature vectors. Each sequence is 51 cells long corresponding to 51 frames of each video. Each sequence has 1024 features that are used as input to the LSTM network.

### 4.2.5 BiLSTM Network Training

The next step is to build LSTM network to classify gas leaks. The sequences of feature vectors are passed to the Sequence Input Layer with 1024 nodes. Input layer is connected to a bidirectional long short-term memory layer (BiLSTM) with 2000 units and a 50% dropout layer. BiLSTM is the learning mechanism for classification of sequential data that can account for the complete time series at each time step. A dropout layer removes a portion of hidden layer units after each repetition during training and to prevent overfitting of the model (Srivastava et al., 2014).

A fully connected layer with an output size of 5 cells corresponding to gas leak size. SoftMax layer and classification layer are used to assign probabilities of each class and subsequently make a decision (Goodfellow et al., 2016). In this work, we classify video footage into 5 categories:

- No leak, 0 m/s
- Leak, 10 m/s
- Leak, 20 m/s
- Leak, 30 m/s
- Leak, 50 m/s

The original paper was using leakage velocity of 15 m/s. We chose different velocities arbitrarily to showcase different accident scenarios.

Additional networks settings include: minibatch size of 16; adaptive moment estimation (Adam) solver; learning rate increasing from 0.0001 to 2.0; validation as well as data

shuffling after each epoch. Minibatch size affects the number of times the network is updated in an epoch. Minibatch size and learning rate can be adjusted if needed for convergence purposes. Minibatch size also reduces memory requirements since the algorithm must perform calculations only over a fraction of the dataset at a time.

The accuracy of classification is then measured by counting the number of correct predictions and dividing by the total number of cases:

$$Accuracy = \frac{\#(pred == true)}{\# true}$$

A confusion matrix is then constructed to provide visual representation of the results. It compares the number of predicted and true samples of all categories in a single figure (Ting, 2017). The network was then trained and tested for different combinations of time durations to see the flexibility of the model in a context of limited data.

## 4.3 Results

### 4.3.1 Data generation and preparation

As mentioned before, we generated concentration profiles of $CH_4$ after 50 seconds of a gas leak in the receiving terminal. Most setting were unchanged from the previous paper of this simulation published by the original authors, such as ambient temperature, wind velocity and direction, vessel pressure and so on. The only variable part was wind velocity, and turbulence parameters, accordingly. Another choice was regarding elevation selection. The original paper documented concentration profiles at three different heights: 20 meters, 32 meters and 40 meters. All elevations were shown to produce results in accordance with test data, therefore any of the three could be chosen

in this work. The decision to choose 20 meters was made to include as many obstacles as possible, as going higher leaves only tall vessels in cross section. In the future work, it is suggested to generate data from different elevations and comparing the results of gas leak detection between them, and possibly combining the data from different heights for further analysis as a more complete picture.

Simulations were run 25 times for each wind velocity, so a total of 125 times. Since OpenFOAM was used inside a virtual machine, the computational capacity was limited to the computational capacity of the VM. Setting up the model on a separate workstation would take a long time and require assistance from the original authors. In addition, at a time of conducting this research, access to a powerful workstation was denied due to lockdown. As a result, data was generation and analyzed on a laptop, which further limited the computational speed and capacity. Considering all limitations, a single simulation of a gas leak would take approximately 25-40 minutes of calculations. In addition, using a separate software (ParaView), the results would need to be displayed and formed according to a template to avoid any noise during training. This and saving of the results would take additional 3-5 minutes. As it was not possible to compute several simulations at the same time, this was the lengthiest part of the project.

As there were five different leak velocities, we additionally changed velocities slightly to avoid the same results being produced multiple times. For example, in a case of 30 m/s leak velocity, we would choose random velocities in a range from 29.0 to 31.0 for both training and testing purposes. The velocity was not changed for 0 m/s (no leak case). Turbulence parameters were chosen according to an online calculator that was also used by the original authors (IChrome, 2016). These parameters are shown in Table

4.1. The reference length was chosen as 8 meters and kinematic viscosity *v* of the natural gas taken as 1.70e-5 $m^2/s$.

Table 4.1. Turbulence parameters for data generation for different velocities (IChrome, 2016).

| Velocity *v*, m/s | Turbulence kinetic energy *k*, J/kg | Specific dissipation rate $\omega$, 1/s |
|---|---|---|
| 10 m/s | 0.082 | 0.936 |
| 20 m/s | 0.277 | 1.717 |
| 30 m/s | 0.563 | 2.448 |
| 50 m/s | 1.378 | 3.828 |

The data was saved in folders in a form of sets of images (.png), 51 image for each simulation. When the integrity of all files was checked, the complete visual dataset was moved outside of the virtual machine and saved on the laptop.

Then, using MATLAB *writeVideo* function the sets of images were transformed into videos, one video per one set (folder). Since there were 51 frames per simulation, we choose a framerate of 17 to make 3 second clips. The videos were in the same format as the images, 759 by 610 pixels. These videos were then resized to match the GoogLeNet neural network input specifications of 224 by 224 pixels. Alternatively, one could resize the images first, and then compile them into videos. Due to arrangement of images into different folders, it would take too long to manually resize

all images, therefore we produced videos first and then worked with them (only 125 videos to be resized).

It should also be noted that all videos were 3 seconds long and that simplified the data preparation process significantly. In case the available data is varying duration, an additional step is necessary to remove longer sequences. Having large time diversity may lead to reduction in classification accuracy, therefore outliers should be either removed or trimmed down.

### 4.3.2 Extraction of feature vectors and network training

When the videos are prepared, they can be used to extract sequences of feature vectors. The *activations* function of the GoogLeNet returns feature vectors of each frame of the video, which are saved separately. The output of the *activations* produces 1024 by 51 cell array for each video, where 51 is the number of frames and 1024 is the number of features.

The sequences of feature vectors are then used as a sequence input layer to train the bidirectional long short-term memory network to classify the sequences into gas leak categories. 20% of the training dataset was used for validation. The training procedure achieved 100% validation accuracy by $6^{th}$ epoch. The trained network is then used to classify testing samples and the results are shown in a form of confusion matrix in Figure 4.5.

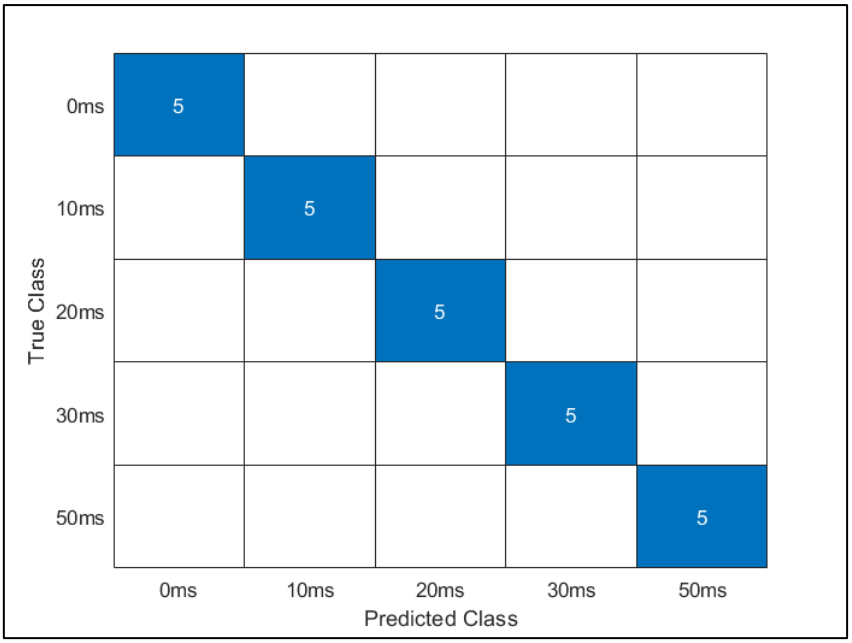



Figure 4.5 Confusion matrix for gas leak size classification at t = 50s

The trained network correctly classified 25 videos of gas leaks into correct categories.

Finally, the number of frames that presented to the network for training was limited to see how reduced information affects classification accuracy. Effectively, we were looking at the prediction accuracy at different times to see at what point can the model correctly identify leaks. We set a minimum time of 5 seconds and train-tested the model repeatedly until t = 50 s. Classification accuracy over time is show in Figure 4.6. At t = 27 s, the accuracy became 100% and we noticed a steady reduction in epochs needed for network training, as expected. However, in this case the network was trained and tested using the same time. Which means that in this situation it would be enough to gather data of the first 27 seconds.

On the other hand, this work investigated whether the model that was trained on a full 50-second data could be applied to a limited time testing data. In other words, how well

would a complete model be able to predict incomplete data. Figure 4.7 shows the accuracies from 10 seconds to 50 seconds. The data suggests that applying the full model is unreliable for the first 26 seconds; the accuracy only starts to increase at t = 27 seconds. However, the results don't get significantly better until t = 39 seconds, where classification accuracy reaches 84%, and for t > 39 it becomes a confident 100%. Therefore, applying a complete model to incomplete dataset can be allowed only for situations where at least 40 seconds elapsed after the leakage.

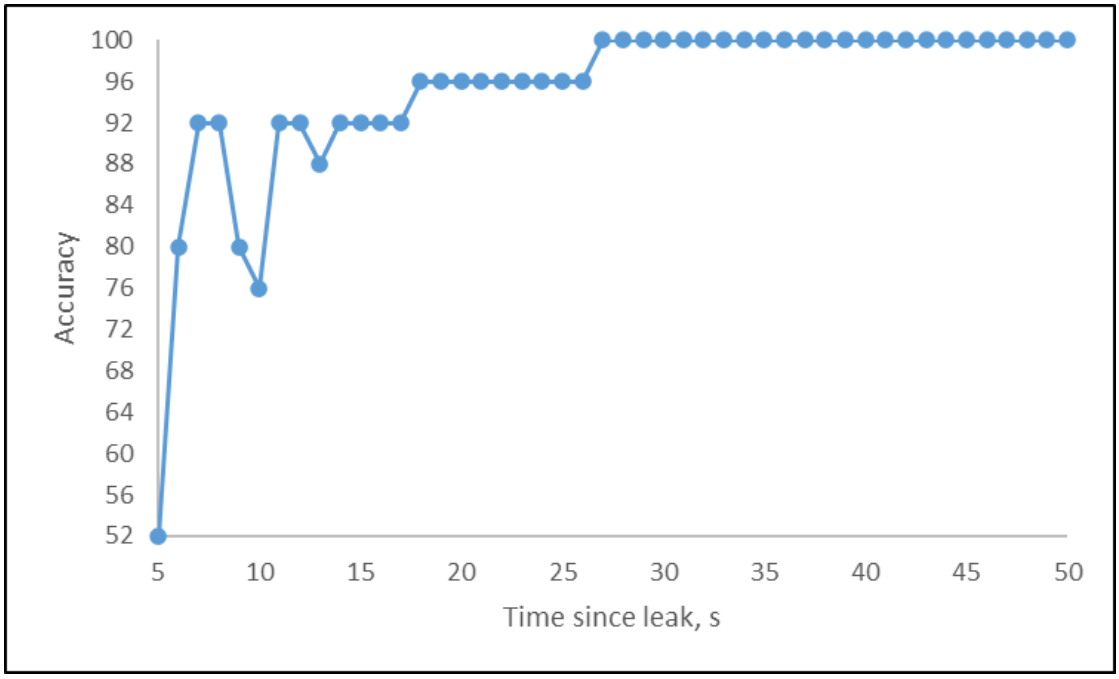



Figure 4.6 Accuracy of gas leak classification using the same duration of training and testing data

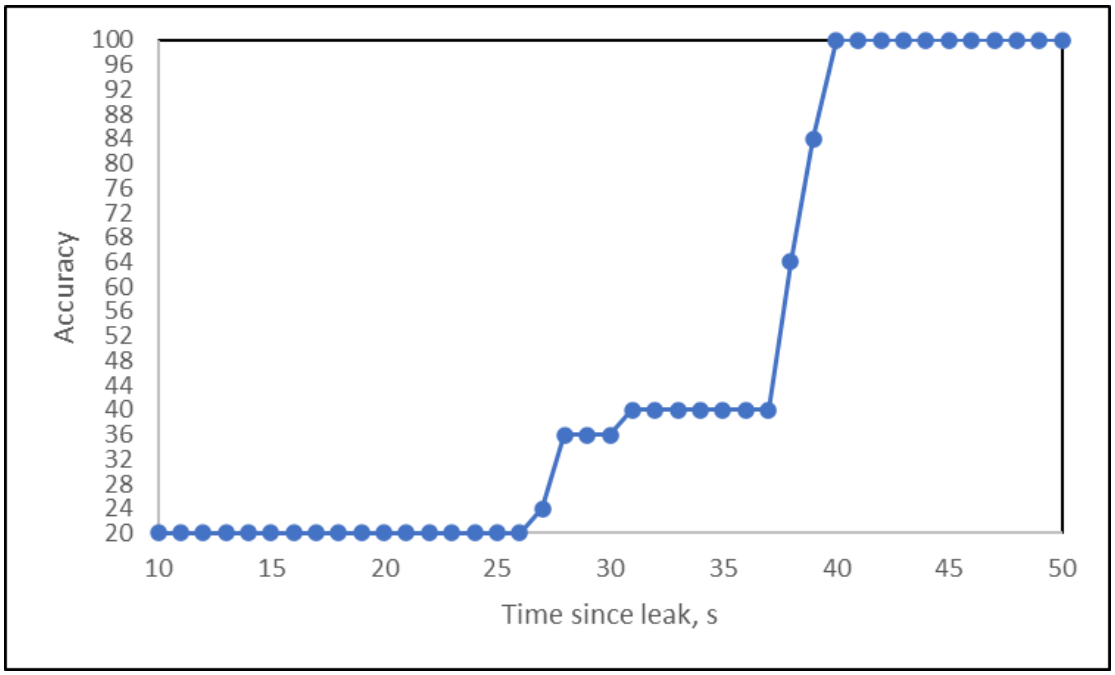



Figure 4.7 Accuracy of gas leak classification using a 50 second-trained model on limited duration testing data

Analyzing confusion plots, we found that for lower times the model would predict a "no leak" result. The model would underpredict rather than overpredict, meaning there were no false positives throughout simulations. As time increased, the model started to detect leaks, but would often misclassify their magnitudes.

### 4.4 Discussion

The results of the initial test run are shown in Figure 4.5. The model was successful in classifying leak sizes based on provided sequences of concentration profiles over a period of 50 seconds since leak initiation. This accuracy was achieved partially due to homogenous nature of data: it was generated via CFD simulation, which did not account for numerous sources of noise. If real data were used, it is possible the prediction accuracy would decline. However, since the model was proven to be in accordance with

testing on a real receiving terminal site, we can conclude that the model's performance is relevant. It is also important to note that only 100 sets of data were used for training of the model, which only accounted for one factor of the gas leak phenomenon – gas leak velocity. It is important to test the model's flexibility when it comes to other factors, such as leak location, vessel pressure, ambient conditions etc. The model was able to successfully isolate one parameter and make accurate predictions. The leak velocity is proportional to the size of the formed gas cloud, which is directly linked to potential damage that can result from said leak. If the model can identify the severity of the accident, it can result in a more educated responses to avoid further escalation and loss. In addition, this model can be incorporated in a larger risk assessment system, which would use the leak size to determine the severity of an accident.

Another matter to consider is the type of data used for prediction. We used concentration profiles which are generated via CFD software. While it is a great leak indicator, this type of data may not be readily available at any chemical plant. The model needs to be tested with other types of input, such as infrared thermal imaging, gas sensor data from various locations, and mixed data. Since ConvNet is used for feature extraction, it is advised to utilize visual data if possible.

Although the model performed with absolute accuracy on a 50 second simulated experiment, we wanted to see the flexibility of the model. First, we decided to reduce the amount of data that we feed into the model by reducing the time from leak to detection. For that, we were gradually removing one frame at a time and were re-training the model to see the change in performance. As Figure 4.6 shows, the model retained its performance to a point where only 26 seconds were used. The model was

still able to identify leaks but would struggle to correctly classify its size. As the available data kept decreasing, the prediction accuracy also decreased, and below 10 seconds it would produce undesirable results. This suggests that with proper tuning, the model may need less time than 50 seconds to accurately predict leak and its size.

While this leads to a quicker response, we wanted to see if a trained model could make prediction based on a limited amount of data. Even if the model was trained to detect a leak based on 50 seconds of data, the model should also make predictions if the leakage has not reached a 50 second timer. Figure 4.7 shows the change in performance when a model was trained on a full 50 seconds of data and was forced to make a prediction early. The results suggest that this model does not perform well unless it is fed at least 40 seconds of data. In comparison to the previous analysis, it would be more proficient to set up a gas leak detection model that is trained only on 30 seconds available data, and let it decide based on full 30 seconds, rather than setting up a 50-seconds trained model which is only able to predict after 40 seconds. This suggests that there is a certain amount of optimization that can be done to improve the model.

Aside from model optimization, the model needs to be validated by data acquired from a test site in addition to the fact that the CFD model used to generate data was proven legitimate. A different geometrical setup should also be tried, as well as different leak locations need to be tested. The ability of the model to predict leak location and its size needs to be examined. Finally, the effect of noise should be investigated on the model's performance. In particular, a potential for discerning leak over noise is a major concern when it comes to industrial application and should be a focused of the future work.

## 4.5 Conclusion

In this paper we developed a model to detect and classify gas leak based on series of concentration profiles. This work presents a novel approach to gas leak detection with neural networks based on temporal data. The model consists of feature extraction element and classification element. To extract features from visual data we used GoogLeNet pretrained convolutional network that converted input video files into sequences of feature vectors; classification was performed using a bidirectional long short-term memory layer neural network. The data was generated using a hybrid model that incorporates CFD and the ensemble Kalman filter and was shown to produce results comparable to field tests. The major advantage of the proposed method is that it does not require fine tuning of layer configuration and other training parameters to produce satisfying results.

The model had 100% accuracy when trained and tested on a 50-second dataset. The generated input data lacked the presence of noise and it only examined one parameter of leakage – its size. And while the leak size can be directly correlated to risk level, other factors should also be considered, such as leak location, variation of ambient parameters, gas temperature and pressure and so on. We then examined classification accuracy based on shortened dataset. For the receiving terminal used in our case, as few as 27 seconds of footage were enough for the model to accurately classify faults. We further investigated model's ability to make predictions based on limited information. As such, we fed a model trained on 50 seconds of data testing samples that contained shortened datasets. The results show that the ability to classify leakage was reduced significantly and the model produced acceptable results only when it was fed almost

entire dataset (40+ seconds). Under these circumstances, the model did not produce false negatives and mostly suffered in its ability to classify detected leaks. These findings suggest that a lot of optimizations can be done aside from training the neural network. The model can be tuned to detect and classify gas leaks in a short period of time with high accuracy, which can result in timely response to an accident, stop escalation and reduce damage.

The model needs further testing incorporating field data. In addition to testing more gas leak factors, other type of visual data needs to be used, such as thermal imaging. Incorporation of sensor data and its effect on prediction accuracy should be investigated to improve the model efficiency. Finally, GoogLeNet needs to be compared to other pretrained convolutional networks to determine the most fitting one for this type of input data.

**Reference list**

Bonvicini, S., Antonioni, G., Morra, P., & Cozzani, V. (2015). Quantitative assessment of environmental risk due to accidental spills from onshore pipelines. *Process Safety and Environmental Protection*, *93*, 31–49. https://doi.org/https://doi.org/10.1016/j.psep.2014.04.007

Datta, S., & Sarkar, S. (2016). A review on different pipeline fault detection methods. *Journal of Loss Prevention in the Process Industries*, *41*, 97–106. https://doi.org/https://doi.org/10.1016/j.jlp.2016.03.010

Doshmanziari, R., Khaloozadeh, H., & Nikoofard, A. (2020). Gas pipeline leakage detection based on sensor fusion under model-based fault detection framework. *Journal*

*of Petroleum Science and Engineering*, *184*, 106581. https://doi.org/https://doi.org/10.1016/j.petrol.2019.106581

Eckerman, I. (2005). *The Bhopal Saga - Causes and Consequences of the World's Largest Industrial Disaster*. Universities Press, India.

Fahimipirehgalin, M., Trunzer, E., Odenweller, M., & Vogel-Heuser, B. (2021). Automatic Visual Leakage Detection and Localization from Pipelines in Chemical Process Plants Using Machine Vision Techniques. *Engineering*, *7*(6), 758–776. https://doi.org/https://doi.org/10.1016/j.eng.2020.08.026

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Softmax Units for Multinoulli Output Distributions. In *Deep Learning* (pp. 180–184). MIT Press. http://www.deeplearningbook.org

IChrome. (2016). *Turbulence Calculator*. http://ichrome.com/blogs/archives/342

Jain, S. (2017). *Data Augmentation | How to use Deep Learning when you have Limited Data*. https://medium.com/nanonets/nanonets-how-to-use-deep-learning-when-you-have-limited-data-f68c0b512cab

Meribout, M., Khezzar, L., Azzi, A., & Ghendour, N. (2020). Leak detection systems in oil and gas fields: Present trends and future prospects. *Flow Measurement and Instrumentation*, *75*, 101772. https://doi.org/https://doi.org/10.1016/j.flowmeasinst.2020.101772

Ning, F., Cheng, Z., Meng, D., Duan, S., & Wei, J. (2021). Enhanced spectrum convolutional neural architecture: An intelligent leak detection method for gas pipeline.

*Process Safety and Environmental Protection*, *146*, 726–735. https://doi.org/https://doi.org/10.1016/j.psep.2020.12.011

Pérez-Pérez, E. J., López-Estrada, F. R., Valencia-Palomo, G., Torres, L., Puig, V., & Mina-Antonio, J. D. (2021). Leak diagnosis in pipelines using a combined artificial neural network approach. *Control Engineering Practice*, *107*, 104677. https://doi.org/https://doi.org/10.1016/j.conengprac.2020.104677

Song, D., Lee, K., Phark, C., & Jung, S. (2021). Spatiotemporal and layout-adaptive prediction of leak gas dispersion by encoding-prediction neural network. *Process Safety and Environmental Protection*, *151*, 365–372. https://doi.org/https://doi.org/10.1016/j.psep.2021.05.021

Song, S. J., & Jang, Y. G. (2018, September 21). Construction of digital twin geotechnical resistance model for liquefaction risk evaluation. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3284557.3284739

Song, Y., & Li, S. (2021). Gas leak detection in galvanised steel pipe with internal flow noise using convolutional neural network. *Process Safety and Environmental Protection*, *146*, 736–744. https://doi.org/https://doi.org/10.1016/j.psep.2020.11.053

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Venhoucke, V., & Rabinovich, A. (2015). Going Deeper with Convolutions. *CVPT2015*.

https://static.googleusercontent.com/media/research.google.com/ru//pubs/archive/43022.pdf

Ting, K. M. (2017). Confusion Matrix. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of Machine Learning and Data Mining* (p. 260). Springer US. https://doi.org/10.1007/978-1-4899-7687-1_50

Travis, B., Dubey, M., & Sauer, J. (2020). Neural networks to locate and quantify fugitive natural gas leaks for a MIR detection system. *Atmospheric Environment: X*, *8*, 100092. https://doi.org/https://doi.org/10.1016/j.aeaoa.2020.100092

Vollmer, M., & Möllmann, K. P. (2017). *Infrared thermal imaging: fundamentals, research and applications*. John Wiley & Sons.

Wang, W., Mao, X., Liang, H., Yang, D., Zhang, J., & Liu, S. (2021). Experimental research on in-pipe leaks detection of acoustic signature in gas pipelines based on the artificial neural network. *Measurement*, *183*, 109875. https://doi.org/https://doi.org/10.1016/j.measurement.2021.109875

Wu, J., Cai, J., Yuan, S., Zhang, X., & Reniers, G. (2021). CFD and EnKF coupling estimation of LNG leakage and dispersion. *Safety Science*, *139*(June 2020), 105263. https://doi.org/10.1016/j.ssci.2021.105263

Xiao, Q., Li, J., Sun, J., Feng, H., & Jin, S. (2018). Natural-gas pipeline leak location using variational mode decomposition analysis and cross-time–frequency spectrum. *Measurement*, *124*, 163–172. https://doi.org/https://doi.org/10.1016/j.measurement.2018.04.030

# Chapter 5: Conclusions and feature work

## 5.1 Conclusion

Two hybrid neural network-based approaches have been proposed for fault detection and diagnosis. The first hybrid method is a combination of kPCA and DNN. The proposed method combines the nonlinear dimensionality reduction from kPCA and then trains reduced data with deep neural network to detect and diagnose faults. Reduction of data complexity allowed for a better learning process of the neural network. Kernel PCA was employed specifically to tackle complex process data. The model was tested on a benchmark Tennessee Eastman Process. The model also explored the possibility of dividing fault detection and diagnosis tasks into separate deep neural networks. This showed to be beneficial in terms of classification accuracy as well as training time.

The second work proposed a hybrid model that consisted of a convolutional network and a BiLSTM network. The feature extraction was performed by the pretrained convolutional network and classification of temporal data was done by BiLSTM network. Convolutional network was a great feature extraction tool for graphical-type input data, and BiLSTM network excelled at classifying gas leaks over 50 seconds. In addition, the model was further optimized by proper selection of the amount of data fed into the model, which resulted in the new model being able to accurately predict gas leaks and their size within first 30 seconds. A strong advantage of this work is that the amount of manual tuning of network parameters was minimal. This model was validated by a simulated data of gas leak in an LNG receiving terminal. The

simulation was previously validated with experimental data, which in turn validates the hybrid leak detection model.

### 5.2 Future work

i. Kernel PCA is not dynamic in nature, and it cannot be used for dynamic FDD. Future work is required to test variations of kPCA in combination of DNN, such as dynamic kPCA.

ii. Kernel PCA is computationally very expensive in both memory and time regard. Modifications of kPCA, such as TP-IKPCA, need to be employed and tested on large datasets.

iii. Future work should focus on building a framework for neural network parameter selection as currently this process is mostly intuitive.

iv. This work explored only supervised ANN-based approaches. Combination of kPCA and DNN for unsupervised dataset should be investigated.

v. GoogLeNet ConvNet was used as a feature extraction tool in the second work. Other pretrained networks should be tested for different input data types.

vi. The models need to be tested with field data to examine the difficulty and the cost of implementation and monitoring.

vii. Future work needs to be conducted to investigate multiple faults/leaks scenarios.

viii. Effect of noisy data should be investigated.

ix. Various visual data should be tested, such as CCTV footage or footage from IR camera.

x. Effect of discerning leak over noise should be investigated.

References

Adedigba, S. A., Khan, F., & Yang, M. (2017). Dynamic Failure Analysis of Process Systems Using Principal Component Analysis and Bayesian Network. *Ind. Eng. Chem. Res.*, *56*(2094–2106).

Ashley, K. D. (2003). Case-Based Reasoning Research and Development: 5th International Conference on Case-Based Reasoning. *ICCBR 2003*.

Azhdari, M., & Mehranbod, N. (2010). Application of Bayesian belief networks to fault detection and diagnosis of industrial processes. *Chemistry and Chemical Engineering (ICCCE)*, 92–96.

Becraft, W. R., Lee, P. L., & Newell, R. B. (1991). Integration of neural networks and expert systems for process fault diagnosis. *Proc. of the 12th IJCAI*, 832–837.

Benkouider, A., Kessas, R., Yahiaoui, A., Buvat, J., & Guella, S. (2012). A hybrid approach to faults detection and diagnosis in batch and semi-batch reactors by using ekf and neural network classifier. *Journal of Loss Prevention in the Process Industries*, *25*(4), 694–702.

Chang, C. T., & Chen, J. W. (1995). Implementation issues concerning the ekf-based fault diagnosis techniques. *Chemical Engineering Science*, *50*(18), 2861–2882.

Che Mid, E., & Dua, V. (2017). Hierarchical decision process for fault administration. *Industrial & Engineering Chemistry Research*, *56*, 8000–8015.

Chen, L. W., & Modarres, M. (1992). Hierarchical decision process for fault administration. *Computers & Chemical Engineering*, *16*, 425–448.

Choi, S., Lee, C., Lee, J., Park, J., & Lee, I. (2005). Fault detection and ident ification

of nonlinear processes based on kernel pea. *Chemometrics and Intelligent Laboratory Systems*, *75*(1), 55–67.

Chow, E., & Willsky, A. S. (1984). Analytical redundancy and the design of robust failure detection systems. *IEEE Transactions on Automatic Control*, *29*, 603–614.

Das, A., Maiti, J., & Banerjee, R. N. (2012). Process monitoring and fault detection strategies: a review. *International Journal of Quality and Reliability Management*, *29*, 720–752.

Dunia, R., Qin, S. J., Edgar, T. F., & McAvoy, T. J. (1996). Identification of faulty sensors using principal component analysis. *AIChE J.*, *42*, 2797–2812.

Frank, P. (1994). On-line fault detection in uncertain nonlinear systems using diagnostic observers: a survey. *International Jpurnal of Systems Science*, *25*(12), 2129–2154.

Frank, P. ., & Wünnenberg, J. (1989). *Fault diagnosis in dynamic systems: theory and applications*. NY: Prentice Hall.

Frank, P. M., & Ding, X. (1997). Survey of rebust residual generation and evaluation methods in observer-based fault detection systems. *J. Process Control*, *7*, 403–424.

Fuan, W., Hongkai, J., Haidong, S., Wenjing, D., & Shuaipeng, W. (2017). An adaptive deep convolutional neural network for rolling bearing fault diagnosis. *Measurement Science and Technology*, *28*(9), 095005.

Ge, Z., Song, Z., & Gao, F. (2013). Review of recent research on data-based process monitoring. *Ind. Eng. Chem. Res.*, *52*(10), 3543–3562.

Gertler, J. (1997). Fault detection and isolation using parity relations. *Control Engineering Practice*, *5*, 653–661.

Gertler, J., & Monajemy, R. (1995). Generating directional residuals with dynamic parity relations. *Automatica*, *31*, 627–635.

Gertler, J., & Singer, D. (1990). A new structural framework for parity equation-based failure detection and isolation. *Automatica*, *26*, 381–388.

Gharahbagheri, H., Imtiaz, S., & Khan, F. (2017). Root Cause Diagnosis of Process Fault Using KPCA and Bayesian Network. *Industrial & Engineering Chemistry Research*, *56*. https://doi.org/10.1021/acs.iecr.6b01916

Gonzalez, R., Huang, B., & Lau, E. (2015). Process monitoring using kernel density estimation and Bayesian networking with an industrial case study. *ISA Transactions*, *58*, 330–347.

Grant, P. W., Harris, P. M., & Moseley, L. G. (1996). Fault diagnosis for industrial printers using case-based reasoning. *Engineering Applications of Artificial Intelligence*, *9*, 163–173.

Guo, X., Chen, L., & Shen, C. (2016). Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement*, *93*, 490–502. https://doi.org/https://doi.org/10.1016/j.measurement.2016.07.054

Himmelblau, D. M. (1978). *Fault detection and diagnosis in chemical and*

*petrochemical processes*. Elsevier Scientific Publishing Company.

Hu, Y., & Yang, Z. (2021). Impact of inter-individual variability on the estimation of default mode network in temporal concatenation group ICA. *NeuroImage*, *237*, 118114. https://doi.org/https://doi.org/10.1016/j.neuroimage.2021.118114

Huang, B. (2008). Bayesian methods for control loop monitoring and diagnosis. *Journal of Process Control*, *18*(9), 829–838.

Iri, M., Aoki, K., O'Shima, E., & Matsuyama, H. (1979). An algorithm for diagnosis of system failures in the chemical process. *Computers & Chemical Engineering*, *3*(1–4), 489–493.

Isermann, R. (1997). Supervision, fault-detection and fault diagnosis methods - an introduction. *Control Engineering Practice*, *5*(5), 639–652.

Isermann, R. (2006). *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media.

Joe Qin, S. (2003). Statistical process monitoring: basics and beyond. *Journal of Chemometrics*, *17*, 480–502.

Kamatchi Kannan, V., Srimathi, R., Gomathi, V., Valarmathi, R., & PrithiEkammai, L. T. (2021). Investigation of unknown input observer for sensor fault diagnosis for a CSTR process. *Materials Today: Proceedings*, *45*, 3431–3437. https://doi.org/https://doi.org/10.1016/j.matpr.2020.12.931

Kelion, L. (2021). Factory blaze adds to computer chip supply crysis. *BBC Tech*.

Khakzad, N., Khan, F., & Amyotte, P. (2013). Dynamic safety analysis of process

systems by mapping bow-tie into bayesian network. *Process Safety and Environmental Protection*, *91*(1–2), 46–53.

Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artificial Intelligence Review*, *6*, 3–34.

Kourty, T., & MacGregor, J. F. (1995). Process analysis, monitoring and diagnosis, using multivariate projection methods. *Chemometrics and Intelligent Laboratory Systems*, *28*, 3–21.

Ku, W., Storer, R. H., & Georgakis, C. (1995). Disturbance detection and isolation by dynamic principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *30*(1), 179–196. https://doi.org/10.1016/0169-7439(95)00076-3

LaViola, J. J. (2003). A comparison of unscented and extended Kalman filtering for estimating quaternion motion. *American Control Conference*, 2435–2440.

Mah, R., & Tamhane, A. (2004). Detection of gross errors in process data. *AIChE J.2*, *28*(5), 828–830.

Moore, K. S. (2021). How and when chip shortage will end. *IEEE Spectrum*.

Musharraf, M., Hassan, J., Khan, F., Veitch, B., MacKinnon, S., & Imtiaz, S. (2013). Human reliability assessment during offshore emergency conditions. *Safety Science*, *59*, 19–27. https://doi.org/https://doi.org/10.1016/j.ssci.2013.04.001

Mylaraswamy, D., & Venkatasubramanian, V. (1997). A hybrid framework for large scale process fault diagnosis. *Computers & Chemical Engineering*, *21*, 935–940.

Nan, C., Khan, F., & Iqbal, M. T. (2008). Real-time fault diagnosis using knowledge-

based expert system. *Process Safety and Environmental Protection*, *86*(1 B), 55–71. https://doi.org/10.1016/j.psep.2007.10.014

Neapolitan, R. E. (2004). *Learning bayesian networks*. Pearson Prentice Hall Upper Saddle River.

Nimmo, I. (1995). Adequately address abnormal operations. *Chem. Eng. Prog.*, *91*, 36–45.

Ning, F., Cheng, Z., Meng, D., Duan, S., & Wei, J. (2021). Enhanced spectrum convolutional neural architecture: An intelligent leak detection method for gas pipeline. *Process Safety and Environmental Protection*, *146*, 726–735. https://doi.org/https://doi.org/10.1016/j.psep.2020.12.011

Odendaal, H. M., & Jones, T. (2014). Actuator fault detection and isolation: An optimised parity space approach. *Control Engineering Practice*, *26*, 222–232.

Patton, R. J., & Chen, J. (1991). A review of parity space approaches to fault diagnosis. *IFAC/IMACS Safeprocess Conference*, 65–81.

Pérez-Pérez, E. J., López-Estrada, F. R., Valencia-Palomo, G., Torres, L., Puig, V., & Mina-Antonio, J. D. (2021). Leak diagnosis in pipelines using a combined artificial neural network approach. *Control Engineering Practice*, *107*, 104677. https://doi.org/https://doi.org/10.1016/j.conengprac.2020.104677

Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., & Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nature Genetics*, *38*, 904–909.

Severson, K., Chaiwatanodom, P., & Braatz, R. D. (2016). Perspectives on process monitoring of industrial systems. *Annual Reviews in Control*.

Shao, H., Jiang, H., Zhang, H., Duan, W., Liang, T., & Wu, S. (2018). Rolling bearing fault feature learning using improved convolutional deep belief network with compressed sensing. *Mechanical Systems and Signal Processing*, *100*, 743–765. https://doi.org/https://doi.org/10.1016/j.ymssp.2017.08.002

Shaw, V. (2021). Silicone fab explosion in Xinjiang threatens further poly shortage. *PV Magazine*.

Song, D., Lee, K., Phark, C., & Jung, S. (2021). Spatiotemporal and layout-adaptive prediction of leak gas dispersion by encoding-prediction neural network. *Process Safety and Environmental Protection*, *151*, 365–372. https://doi.org/https://doi.org/10.1016/j.psep.2021.05.021

Song, Y., & Li, S. (2021). Gas leak detection in galvanised steel pipe with internal flow noise using convolutional neural network. *Process Safety and Environmental Protection*, *146*, 736–744. https://doi.org/https://doi.org/10.1016/j.psep.2020.11.053

Sorsa, T., & Koivo, H. N. (1993). Application of artificial neural networks in process fault diagnosis. *Automatica*, *29*(4), 843–849. https://doi.org/https://doi.org/10.1016/0005-1098(93)90090-G

Sotomayor, O. A. Z., & Odloak, D. (2005). Observer-based fault diagnosis in chemical plants. *Chemical Engineering Journa*, *112*, 93–108.

Staroswiecki, M., Cassar, J. P., & Cocquempot, V. (1993). Generation of optimal

structured residuals in the parity space. *IFAC Proceedings Volume 26*, 535–542.

Travis, B., Dubey, M., & Sauer, J. (2020). Neural networks to locate and quantify fugitive natural gas leaks for a MIR detection system. *Atmospheric Environment: X*, *8*, 100092. https://doi.org/https://doi.org/10.1016/j.aeaoa.2020.100092

Umeda, T., Kuriyama, T., O'Shima, E., & Matsuyama, H. (1980). A graphical approach to cause and effect analysis of chemical processing systems. *Chemical Engineering Science*, *35*(12), 2379–2388.

Venkatasubramanian, V., Rengaswamy, R., & Kavuri, S. N. (2003). A review of process fault detection and diagnosis: Part II: Qualitative models and search strategies. *Computers & Chemical Engineering*, *27*(3), 313–326. https://doi.org/https://doi.org/10.1016/S0098-1354(02)00161-8

Venkatasubramanian, V., Rengaswamy, R., Yin, K., & Kavuri, S. N. (2003a). A review of fault detection and diagnosis. Part III: Process history based methods. *Computers and Chemical Engineering*, *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. *Computers & Chemical Engineering*, *27*(3), 327–346. https://doi.org/10.1016/s0098-1354(02)00162-x

Vesely, W. E., Goldberg, F. F., Roberts, N. H., & Haasl, D. F. (1981). *Fault tree handbook*. Nuclear Regulatory Commission Washington dc.

Wan, E. A., & Vam der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. *Adaptive Systems for Signal Processing, Communications, and*

*Control Symposium 2000*, 153–158.

Wang, W., Mao, X., Liang, H., Yang, D., Zhang, J., & Liu, S. (2021). Experimental research on in-pipe leaks detection of acoustic signature in gas pipelines based on the artificial neural network. *Measurement*, *183*, 109875. https://doi.org/https://doi.org/10.1016/j.measurement.2021.109875

Wilcox, N. A. (1992). *The possible-cause effects graph model of process fault diagnosis*. The University of Texa, Austin.

Wilcox, N. A., & Himmelblau, D. M. (1994). The possible cause and effect graphs (PCEG) model for fault diagnosis—I. Methodology. *Computers & Chemical Engineering*, *18*(2), 103–116. https://doi.org/https://doi.org/10.1016/0098-1354(94)80131-2

Willsky, A. S. (1976). A survey of design methods for failure detection in dynamic systems. *Automatica*, *12*(6), 601–611.

Wilson, A. G., & Huzurbazar, A. V. (2007). Bayesian networks for multilevel system reliability. *Reliability Engineering and System Safety*, *92*(10), 1413–1429.

Woodward, J. L., & Pitbaldo, R. (2010). *NG risk based safety: modeling and consequen ce analysis*. John Wiley & Sons.

Yang, F., Shah, S., & Xiao, D. (2012). Signed directed graph based modeling and its validation from process knowledge and process data. *International Journal of Applied Mathematics and Computer Science*, *22*, 41–53.

Yang, F., Xiao, D., & Shah, S. (2010). Qualitative fault detection and hazard analysis

based on signed directed graphs for large-scale complex systems. In *Fault Detection*. IntechOpen.

Young, P. (1981). Parameter estimation for continuous-time models. *Automatica*, *17*, 23–39.

Yu, D., Shields, D. N., & Daley, S. (1996). A hybrid fault diagnosis approach using neural networks. *Neural Computing and Applications*, *4*, 21–26.

Yu, H., Khan, F., & Garaniya, V. (2015). Modified Independent Component Analysis and Bayesian Network- Based Two-Stage Fault Diagnosis of Process Operations. *Industrial & Engineering Chemistry Research*, *54*, 1–19. https://doi.org/10.1021/ie503530v

Zadakbar, O., Imtiaz, S., & Khan, F. (2012). Dynamic Risk Assessment and Fault Detection Using Principal Component Analysis. *Ind. Eng. Chem. Res.*, *52*, 809–816.

Zeng, G., He, Y., Yu, Z., Yang, X., Yang, R., & Zhang, L. (2016). Preparation of novel high copper ions removal membranes by embedding organosilane-functionalized multi-walled carbon nanotube. *Journal of Chemical Technology and Biotechnology*, *91*(8), 2322–2330. https://doi.org/10.1002/jctb.4820

Zhang, Z., & Zhao, J. (2017). A deep belief network based fault diagnosis model for complex chemical processes. *Computers and Chemical Engineering*, *107*, 395–407. https://doi.org/10.1016/j.compchemeng.2017.02.041

Zhao, H., Liu, J., Dong, W., Sun, X., & Ji, Y. (2017). An improved case-based reasoning method and its application on fault diagnosis of Tennessee Eastman

process. *Neurocomputing*, *249*, 266–276.

Zhao, R., Wang, D., Yan, R., Mao, K., Shen, F., & Wang, J. (2018). Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Transactions on Industrial Electronics*, *65*(2), 1539–1548.

Zhong, M., Song, Y., & Ding, S. X. (2015). Parity space-based fault detection for linear discrete time-varying systems with unknown input. *Automatica*, *59*, 120–126.