

Fault Detection of Supermarket Refrigeration Systems Using Convolutional Neural Network*

1st Zahra Soltani
Dept. Electronic Systems
Aalborg University
Aalborg, Denmark
zns@es.aau.dk

2nd Kresten Kjaer Soerensen
Dept. Transport
Bitzer electronics
Soenderborg, Denmark
Kresten.soerensen@bitzerdk.com

3rd John Leth
Dept. Electronic Systems
Aalborg University
Aalborg, Denmark
jll@es.aau.dk

4th Jan Dimon Bendtsen
Dept. Electronic Systems
Aalborg University
Aalborg, Denmark
dimon@es.aau.dk

Abstract—The functionality of supermarket refrigeration systems (SRS) has a significant impact on the quality of food products and potentially human health. Automatic fault detection and diagnosis of SRS is desired by manufacturers and customers as performance is improved, and energy consumption and cost is lowered. In this work, Convolutional Neural Networks (CNN) are applied for fault detection and diagnosis of SRS. The network is found to be able to classify the fault with 99% accuracy. The sensitivity of the designed model to data quality is also assessed. The results show that the model can classify faults at low sample rates if the training set is large enough. Moreover, the model displays low sensitivity to data quality such as noisy and perturbed validation data, and the frequency of false positives is satisfactorily low as well.

Index Terms—refrigeration, evaporation, fault, classification, machine learning, neural network, convolutional, data quality,

I. INTRODUCTION

The general quality of refrigerated food depends on how accurately its temperature is controlled throughout the cold-chain, from production to the end-users. Improvement of reliability of Supermarket Refrigeration Systems (SRS) by early fault diagnosis is highly relevant when considering the safety of food, human health, and environmental pollution of a large industry. According to [1], because SRS must run night and day, they consume about 50% of the entire energy budget of most supermarkets. Thus, using a faulty refrigeration system can lead to critical economic losses. As a consequence, refrigeration companies try to gain a competitive edge by producing products with as high degree of automation as possible, including for performance monitoring or fault diagnosis.

In this paper, classification of evaporator fan faults is studied; these faults may typically result in inaccurate cooling room temperature, which may lead to food spoilage and energy waste. Therefore, it is of high importance to detect and diagnose evaporator fan faults before they result in damage to the goods. However, constant human monitoring is tedious, expensive and error-prone. Therefore, data-driven Fault Detection and Diagnosis (FDD) has become increasingly popular in the industry, and in particular Artificial Intelligence (AI) is

receiving a lot of attention due to its abilities to make decision instantaneously and deal with vast amounts of data [2].

Convolutional Neural Networks (CNN) is a cornerstone in image processing when it comes to the classification of highly challenging data sets [3]; CNNs are known to be accurate and computationally faster than most other machine learning-based classification methods. In this method, the essential features, information or correlation among data is extracted. Afterwards, the data are classified based on the information. Similar ideas can be used in the classification of signals in signal processing. From this view, CNNs can be used for fault detection and classification.

A number of different data-driven approaches have been proposed. For instance, a combination of a Genetic Algorithm and a Pseudo-inverse matrix algorithm can be found in [4] to obtain parameters and weights of the radial basis function network. This method identifies, successfully, six faults emulated in a laboratory set up. An Extended Kalman Filter (EKF) method is proposed in [5]. Although the EKF performed better and faster fault detection than an ordinary Kalman filter, neither filter was able to distinguish between sensor faults and parametric faults. On the other hand, successful results of Artificial Neural Network (ANN) strategies regarding FDD in chillers have been reported, e.g., [6], and [7]. The Support Vector Data Description (SVDD) method was employed for FDD in chillers in [7], where SVDD is compared with the Principal Component Analysis (PCA) method. The fault detection performance of the SVDD method was better than the PCA method. In [8], refrigerant leaks were detected and diagnosed using a probabilistic ANN algorithm; the ANN could detect refrigerant losses with 90% accuracy. In [9], the better performance of a Probabilistic Neural Network is demonstrated compared to Back Propagation (BP) method as BP has random initial weights leading to a less reliable system. Air handling unit faults are detected using a pattern matching method in [10]. This method is combined with PCA in [11]. It improves the sensitivity of fault detection model and boosts the performance of air handling units.

In image recognition applications CNN is known because of its impressive feature extraction and classification capabilities. These capabilities make it a strong candidate for FDD and process monitoring, where fault patterns might appear in data

This work is partially funded by Innovations fund Denmark and supported by Bitzer electronics A/S, Denmark.

The outline of the rest of the paper is as follows: Section II explains SRS preliminaries, the general methodology of CNN, and its training process. In section III, data collection and CNN design is represented. Different sensitivity tests against data quality are proposed in section IV, and the results are investigated in section V. Finally, conclusions are presented in sections VI.

II. PRELIMINARIES

A. Supermarket refrigeration systems

Refrigeration systems transfer heat in a process where heat is absorbed from a cooling room and released in the ambient environment. During a cycle, the heating absorption and heat dissipation happen by changing the refrigerant phase from liquid to vapour and vapour to liquid, respectively. The nomenclature of the SRS and refrigeration shown in Fig. 1, are introduced in Table I. This system includes two parts; a Bitzer condensing unit and an industrial evaporator for air cooling mounted in an insulated room. The number of evaporators, fans and cooling rooms are different from supermarket to supermarket. In Fig. 1, $Ctrl_{Cond}$ is the condensing unit controller which is connected to the required sensors to control the condenser fan speed and compressor speed V_{cpr} . The compressor speed is controlled to provide the capacity required to keep the temperature in the cooling room. Inside the cooling room, heat is transferred from the goods via the evaporation process to the refrigerant. The evaporator controller is called $Ctrl_{Evap}$ and can be seen in Fig. 1. This controller uses required inputs taken from sensors to control the evaporator fan speed and opening degree of the expansion valve. The opening degree of the expansion valve determines the amount of refrigerant that passes through the evaporator and is controlled to achieve Minimum Stable Superheat (MSS).

The evaporator fan is responsible for circulating air over the evaporator surface and in the cold room to enhance heat

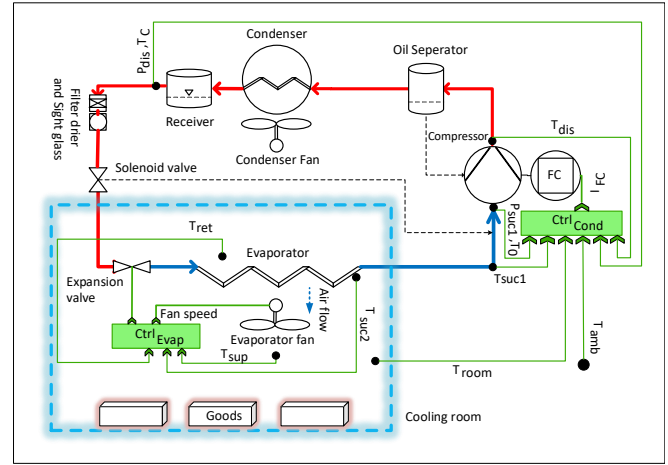


Fig. 1. Schematic of a refrigeration system.

TABLE I
SYMBOLS USED IN THE FIG. 1

Symbols	description	SI unit
T_{room}	cooling room temperature (sensor)	[°C]
T_{amb}	ambient temperature (sensor)	[°C]
$T_{suc1,2}$	suction temperature (sensor)	[°C]
T_0	evaporation temperature (sensor)	[°C]
P_{suc}	suction pressure (sensor)	[Pa]
T_{dis}	discharge temperature (sensor)	[°C]
P_{dis}	discharge pressure (sensor)	[Pa]
T_c	condensing temperature	[°C]
T_{ret}	returned air temperature (Sensor)	[°C]
T_{sup}	supplied air temperature (Sensor)	[°C]
I_{FC}	converter current	[A]
FC	frequency converter	[-]
$Ctrl_{Evap}$	evaporator controller	[-]
$Ctrl_{Cond}$	condensor controller	[-]

transfer into the evaporator and keep an even temperature in the cold room. When the fan does not work or run slowly, there would not be enough airflow around the evaporator pipes. The reduced airflow causes reduced heat transfer. Thus, to compensate and keep the required cooling capacity for the room, C_{trl}^{Evap} increases the temperature difference between the refrigerant and the air. This causes the suction pressure P_{suc} to drop and the vapor density ρ_v at the compressor inlet to decrease as:

$$\rho_v = \frac{P_{\text{suc}} M_m}{RT_0} \quad (1)$$

where M_m is Molar mass and T_0 is evaporation temperature of the refrigerant and R is the ideal refrigerant gas constant. Lower ρ_v leading to more work required by the compressor and increased mass flow rate \dot{m} as:

$$\dot{m} = \rho_v V A \quad (2)$$

where V is the volumetric flow rate of the refrigerant and A is the area of the compressor inlet. Therefore, \dot{m} could be one of the indicators for a faulty evaporator fan. However, it requires knowledge of the compressors parameters, which is not always available. Therefore, the proportion to the mass flow rate is

enough, where A is omitted and the compressor speed (V_{cpr}) is used instead of V . Note that V_{cpr} is proportional to V itself. Thus, a proportional compressor mass flow rate K_{mf} can be used, represented as:

$$K_{\text{mf}} \propto \rho_v V_{\text{cpr}}. \quad (3)$$

In the beginning stage, the evaporator fan fault leads to higher compressor speed and more power consumption. If the compressor reaches its maximum speed due to excessively low P_{suc} the temperature in the cold room will begin to increase. This implies violation of the food quality. Therefore, early FDD is required before any change in the cooling room temperature occurs.

B. CNN methodology

In the sequel, the CNN methodology adopted for fault detection will be presented.

Suppose we are given a set of feature vectors $\{\chi_k\}, k = 1, \dots, K$, each of which belong to a finite set of classes $\{\kappa_n\}, n = 1, \dots, \nu$. The associated *classification problem* is then the challenge of finding a map $\mathcal{N} : \mathcal{X} \rightarrow \{e_n\}, n = 1, \dots, \nu$, where \mathcal{X} is the *feature space* from which χ_k are drawn and $\{e_n\}$ is an orthonormal set of vectors with all entries equal to zero except the j 'th entry, which is one; e_n corresponds to class n .

The map \mathcal{N} will be approximated using a CNN. CNNs are composed of *neurons*, which are nonlinear functions parametrized by so-called *synaptic weights*. The neurons are organized in *layers* – an input layer, several hidden layers and an output layer – and trained using supervised learning. Commonly, CNNs can be decomposed into two separate stages. The first stage, called the feature extraction stage, includes the input layer along with one or more convolutional layers. The second stage includes a number of fully connected layers, which are responsible for the classification – see Fig. 2. The most informative features are collected in the last convolution layer. A flattening layer is the vectorized shape of the last convolutional layer used as an input to the classification stage. The number of neurons of the output layer should match the number of classes ν .

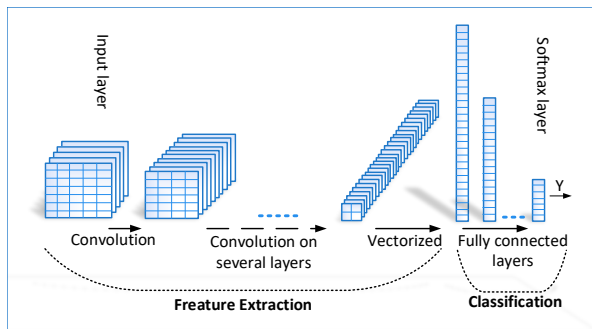


Fig. 2. General design of CNN.

C. Training the model

The first stage of a CNN is organized like a standard multi-layer perceptron network, i.e., all nodes in each layer are feed-forward connected by weights $w_i \in \mathbb{R}$ from inputs $x_i \in \mathbb{R}, i = 1, \dots, n$ via a neuron function $f : \mathbb{R} \rightarrow \mathbb{R}$ to yield a neuron output y :

$$y = f \left(\sum_{i=1}^m w_i x_i - b \right). \quad (4)$$

The input to the first layer is χ_k . The output y from each layer is subsequently used as input for all neurons in the subsequent hidden layer. In neural network terminology, b is called the activation threshold or bias, while the sum of weighted inputs and bias is known as an activation potential, and m is the number of neurons in previous layer. A layer is thus a column vector of neurons, each of which may be parametrized by different sets of weights.

In a CNN, the last layer $F : \mathbb{R}^\nu \rightarrow \mathbb{R}^\nu$ is often chosen as a so-called *Softmax* activation function, whose n 'th component is defined as:

$$\hat{Y}_{k,n} = F_n(y_{k-1,n}) = \frac{\exp(y_{k-1,n})}{\sum_{n=1}^{\nu} \exp(y_{k-1,n})} \quad (5)$$

where $\hat{Y}_k = \hat{Y}_k(W^p)$ is the CNN's estimate of the class of the k 'th feature vector based on the current set of weights W^p . The Softmax function is a smooth approximation to the function $\arg \max(\cdot)$, basically picking out the index n among the entries of the input y_{k-1} with the largest value; that is, $\hat{Y}_k \approx e_n$ if the largest entry in y_{k-1} is found at index n .

To begin training a CNN, it is first necessary to obtain the training data and desired output corresponding to each input. For each class the training data are divided into specific size and each small part is called a mini-batch. The mini-batches stack together at the third dimension. Fig. 3 illustrates the input pre-processing required for training the CNN and the output structure. The CNN needs to learn the corresponding desired output for each input. In addition, for all of the mini-batches in the same class, corresponding outputs are the same and the number of Training mini-batches in each class (T_{mb}) is:

$$T_{\text{mb}} = n_d \left\lceil R \left\lceil \frac{N_s}{S_{\text{mb}}} \right\rceil \right\rceil \quad (6)$$

where n_d is number of data logs, R is split ratio between training and test data, N_s is the number of samples, S_{mb} is the size of each mini-batch. After designing the shape of each class of

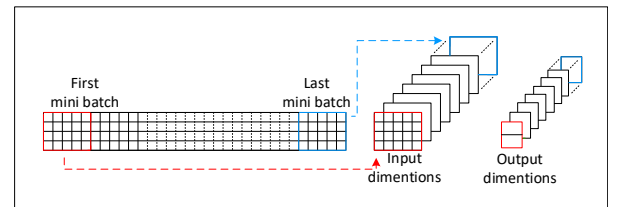


Fig. 3. Input pre-processing and the output structure of each class.

functional and faulty system, The inputs to the CNN require both classes of functional and faulty data. Thus, both classes are concatenated, as shown in Fig. 4. The number of output neurons in the CNN is the same as the number of classes. *Training* is a process in which the network weights are updated

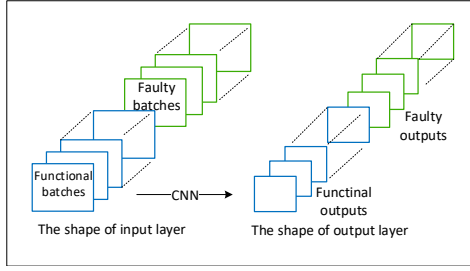


Fig. 4. Visualisation of input and output shape after concatenation of non-faulty and faulty mini-batches.

to give increasingly better predictions of the correct classes as a function of the input feature vectors. Each update of the weights is called an *epoch*. The improvement in prediction is measured by way of a *Loss function*, which should be selected to match the activation function of the output layer; the *cross-entropy* loss function is commonly chosen in classification tasks (as opposed to, for example, the sum-of-squared-error loss function used in function approximation). When only two classes are considered, one may choose a *sigmoid* neuron in the output layer, which always yields an output prediction between zero and one, which may, in turn, be interpreted as a probability of the given feature belonging to the corresponding class. Training with the cross-entropy as the loss function then corresponds to maximizing the conditional log-likelihood of the data being correctly classified as explained in [13].

Given a collection of network weights W^p and ν independent targets (classes), the cross-entropy error for a single example χ_k is given by

$$E_k(\hat{Y}_k, W^p) = -Y_k^\top \ln(\hat{Y}_k) - (\mathbf{1} - Y_k)^\top \ln(\mathbf{1} - \hat{Y}_k). \quad (7)$$

where $\mathbf{1} = [1, 1, \dots, 1]^\top$ and $\ln(\cdot)$ is taken element-wise to yield a ν -dimensional output.

This function estimates the difference between the actual and predicted probability distribution. *Stochastic Gradient Descent* (SGD) optimization is used to tune the weights to improve the prediction—see [14]. The weights in layer l are adjusted in epoch p using

$$w_{il}^{(p)} = w_{il}^{(p-1)} + \alpha \nabla E_k(\hat{Y}_k, W^p) \quad (8)$$

where α is *learning rate* or step size and $\nabla E_k(\hat{Y}_k, W^p)$ is the gradient of the loss function wrt. the weights. We may compute the derivative of the cross-entropy error with respect to each weight connecting the hidden layer neurons to the output layer neurons using the chain rule:

$$\frac{\partial E_k}{\partial w_{n,i}} = \frac{\partial E_k}{\partial \hat{Y}_{k,n}} \frac{\partial \hat{Y}_{k,n}}{\partial u_n} \frac{\partial u_n}{\partial w_{n,i}}$$

where $u_n = \sum_i w_{n,i} x_i - b_n$ is the input to the n 'th neuron in the previous layer. In each epoch, the calculated loss is propagated backward in the network in a layer-by-layer sequential fashion, where the gradients are computed from (4). *Adaptive Moment estimation (Adam)* is a variation of SGD, in which the learning rate α is tuned adaptively to deal with sparse gradient and non-stationary objectives. Moreover, the Adam optimizer is capable of dealing with falling into local minima; see [15] for details.

Convolution of the filters or weights through the feed-forward process prevents having a vast number of weight vector connections in every layer and speed up the network operation. Besides, non-informative features can be eliminated in each layer using the so-called *pooling method*. By using pooling after the convolutional layer, the outputs of each layer are pooled together in the specified filter size. The most common pooling methods are average-pooling, and max-pooling, which collect the average of the outputs, and a maximum of the outputs, respectively.

III. EXPERIMENTS

The condensing unit in the laboratory shown in Fig. 1, consists of a semi-hermetic reciprocating, four-cylinder compressor with a speed range of 25-87 Hz. It has 17KW cooling capacity at 10 °C evaporating temperature using refrigerant R-134a. The two condenser fans have a maximum power consumption of 350 W.

Supermarket condensing units are connected to different evaporation setups, depending on the requirements of each supermarket. Moreover, information from the evaporation part of refrigeration systems is typically not available. In this paper, data is taken from the condensing unit, and the data from the evaporation side is neglected. The evaporator at the Bitzer electronic laboratory has two fans. Fan speeds are controlled by $\text{Ctrl}_{\text{Evap}}$ shown in Fig. 1. In order to emulate the evaporator fan fault, a switch is installed between controller output and relay of one of the fans as seen in Fig. 5. Thus, it is possible to switch on and off, manually, one of the fans and collect data in both conditions.

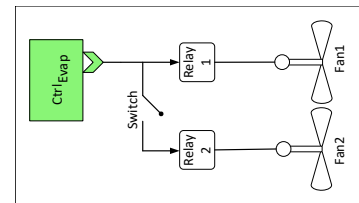


Fig. 5. Stop force on the evaporator fan to emulated defective fan.

A. Data acquisition

Data is taken where the room setpoint varies in the range of 1 °C to 12 °C. Different compressor speeds between 33Hz and 70Hz are applied. This variation is needed in the training phase to learn the response of the system in different states. Moreover, it prevents the over-fitting of the neural network.

When one of the fans is switched off the defective fan fault is emulated on the laboratory set up. Both non-faulty and faulty data are collected with 1 Hz sample rate. Each set of data includes information of P_{suc} , Superheat temperature T_{sh} , V_{cpr} and K_{mf} . These parameters change when the fan is switched off while room temperature remains constant and controlled. When the fault occurs, V_{cpr} increases to compensate P_{suc} and density drop at the inlet of the compressor. This means that the overall efficiency of the system is reduced due to the fault, but, because the system is able to keep the room temperature, this fault would normally not be detected by traditional fault detection. Moreover, K_{mf} is oscillating more because there is less ventilation around the evaporator, and it causes unstable heat transfer around the evaporator. This oscillatory heat transfer induces oscillations in T_{suc} and P_{suc} , which in turn results in K_{mf} oscillating with a higher amplitude. While this fault continues in the system, the cooling room temperature at different locations varies. Finally, due to the lower P_{suc} , and reduced heat transfer, the evaporator surface is colder, and this leads to a faster build-up of ice on the evaporator. Therefore, detecting of accumulated ice is needed more regularly, and this also presents an additional energy cost.

Therefore, a CNN algorithm is used to detect the fault before it affects the room temperature and prevents excessive energy usage due to inefficient running without the fault detection.

B. CNN specification

In this work, six data logs corresponding to various loads and set-points are used. The size of each data log is 4×13000 samples because there are four measurements in each data set, as mentioned in Subsection III-A. When designing CNNs, it is important to select proper hyper-parameters. Hyper-parameters are external and controllable parameters set by the user, including mini-batch size, number of layers, activation functions, filter size, cost function, and optimization method, and so on. In this paper, the mini-batch size is selected as 4×30 samples, which is obtained by manual optimization. The initial learning rate is a key parameter in the training configuration; here α in (8) is chosen as 0.0003. At the output layer a Sigmoid function

$$\hat{Y}_1 = \frac{1}{1 + \exp(y_{k-1})} \quad (9)$$

is used because it is a binary classification. The range of the Sigmoid function is $[0, 1]$, and the classification is performed by a simple threshold; if $\hat{Y}_1 < 0.5$, the class is 0, and 1 otherwise.

In this work, *binary-cross-entropy*

$$E(\hat{Y}_1, W^p) = -Y_1 \ln(\hat{Y}_1) - (1 - Y_1) \ln(1 - \hat{Y}_1) \quad (10)$$

is used as a cost function, which is the same as the cross entropy in (7) for only two output vectors. In (10), $Y_1 = 1$ is the value assigned to class one and $\hat{Y}_1 \in [0, 1]$ is the estimated probability of the input sample belonging to that class. Since

probabilities sum to 1, the second class is assigned the value $Y_2 = 1 - Y_1$ and the corresponding estimate is $\hat{Y}_2 = 1 - \hat{Y}_1$.

In this work, the design of the CNN is improved as table II to obtain better classification results. In this table, S_f stands for the size of the filters, N_f is the number of filters in each layer, *Act* is activation function where ReLU stands for Rectifier Linear Unit, and *MP* is Maxpooling size. Padding type is mentioned as P and valid means that an array of zeroes is applied to the edges of the data when passing through the next layer. The fully connected layer is used with 50% dropout.

TABLE II
DESIGN OF THE CNN ALGORITHM, USED IN THIS WORK.

Layer	S_f	N_f	<i>Act</i>	P	<i>MP</i>
convolution	(2,20)	16	ReLU	valid	(1,3)
convolution	(2,3)	32	ReLU	valid	-
convolution	(1,3)	64	ReLU	valid	-
Flatten	-	-	-	-	-
FC	40	-	ReLU	-	-
Dropout(0.5)	-	-	-	-	-
FC	2	-	Sigmoid	-	-

Dropout is an efficient solution to prevent over-fitting. In this method, the number of neurons is regularized in each layer, and the rest of the neurons are dropped temporarily together with all the inputs and outgoing connections [16].

IV. SENSITIVITY TO DATA QUALITY

As SRS operations and configurations vary significantly due to the individual supermarket's demands and geographical conditions, it is very important that the FDD model has low sensitivity to variations in the data available. Here, a number of experiments is introduced to examine the sensitivity to data quality and reliability of the model. For all experiments, the structure of the model and the hyper-parameters of the CNN model are the same as specified in the table II.

A. Low resolution data

In SRS data acquisition, the sample rate varies between 1 to 0.0003 Hz depends on the embedded hardware memory. In this work, data is re-sampled from 1 Hz to 0.16, 0.016, and 0.0016 Hz. Even though it was of interest to observe the result with much lower sample rates such as 0.0003 Hz, it is not possible to do this here, due to limitations on data log length. One non-faulty and one faulty data log with regards to four different sample rates are introduced in Fig. 6. From top to bottom, the sample rate is decreased by means of down-sampling; these re-sampled data sets will be used for training CNN models. As can be seen, lowering the sample rates from 1 to 0.016 Hz does not change the main features of the data due to the slow dynamics of the refrigeration system. However, at 0.0016 Hz the important features can no longer be detected in the sampled data.

As the data is down-sampled, the number of samples decreases. Therefore, another experiment is done to train the model with the same data length to compare the down-sampling result with the same size of data. The results of these experiments are presented in subsection V-A.

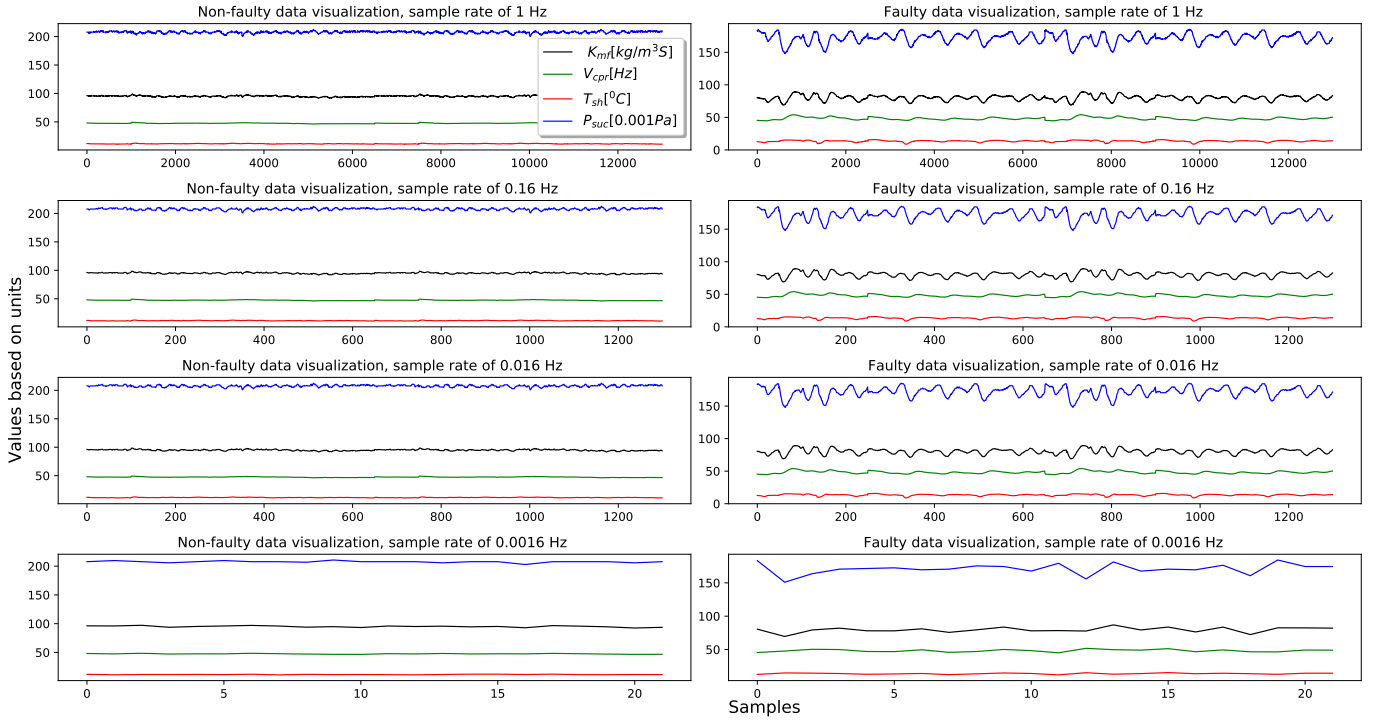


Fig. 6. Visualisation of data re-sampling for Non-faulty data to the left and faulty data to the right.

B. Noisy data

In industrial applications, data can be noisy and incomplete due to different reasons, for instance, sensor noise, electromagnetic propagation, sample dropouts and so forth. In this work, noise is added to data, not only to observe noisy data response but also to generate random new validation data where correlation among parameters is preserved. Random variables with normal distribution $\mathcal{N}(0, 2)$ is selected for this purpose. To do this, the noise S_n is added to the saturation temperature T_{sat} and consequently to T_{sh} as:

$$T_{sh} + S_n = (T_{suc} - T_{sat}) + S_n \quad (11)$$

where T_{suc} is suction temperature or actual temperature of refrigerant after evaporation. Therefore, by any change in T_{sh} , P_{suc} and eventually K_{mf} would change as the correlations are introduced in (1) – (3).

In Fig. 7, Random noise with $\mathcal{N}(0, 2)$ is added to T_{sh} , and eventually, P_{suc} and K_{mf} . In order to test the reliability of the algorithm, noisy validation data is generated 100 times and passed through the network as new validation data sets. The result of this test can be found in subsection V-B.

C. Operating point change

The SRS may operate in different operating points depending on the needed capacity, the layout of the system and the ambient conditions. To the CNN, this will look like offsets or perturbations in measurements that are correlated according to the physics of the system. Thus, a random offset value is applied to T_{sh} on validation data set using random

numbers between $[-3, 3.5]$. In accordance with (1) – (3), P_{suc} and K_{mf} change correspondingly; as P_{suc} should be in its valid refrigeration cycle envelope, the random offset value can not be outside specific ranges when using the available data. This random perturbation is applied 100 times to observe the reliability of the model when the correlation between parameters is different from what is used in the training data.

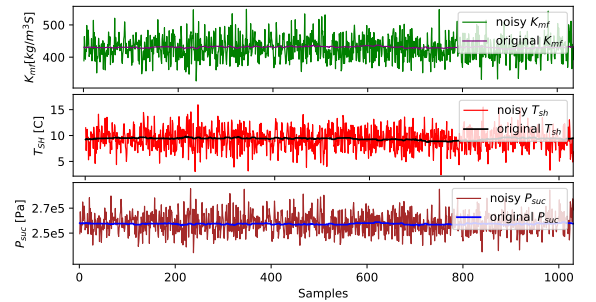


Fig. 7. Example of original and noisy data where noise is added to T_{sh} .

V. RESULTS

A number of different experiments to evaluate the sensitivity of the CNN model were proposed in the previous section. The results of each experiment are presented in the following.

A. Data re-sampling result

Re-sampling of the training data is done as specified in subsection IV-A. Fig. 6 illustrates that even when the sampling

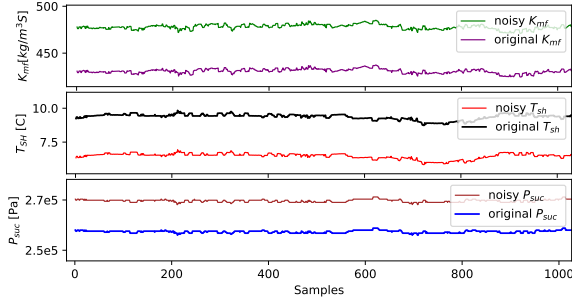


Fig. 8. Example of original and perturbed data.

rate is reduced and the number of samples is lowered, the main features in the data are preserved due to the slow dynamic of SRS. Fig. 9 shows the accuracy of classification and loss function with different sample rates. It is seen that lowering the sample rate also lowers the accuracy of the training process and causes the training to require more iterations. However, it should be noted that the decreased accuracy is not so much due to the lower sample rate itself, but rather due to the lower number of samples available to the CNN. In Fig. 6, down-sampling is continued until 0.0016 Hz in order to obtain a lower bound on the sampling rate; however, with the few data points remaining, it becomes impossible to train the CNN at this sample rate.

Fig. 9 indicates that if the sample rate is kept constant, the accuracy is decreased when the data is shortened (see the experiment in blue, orange, red and brown). It is remarked that the experiments in orange and green (respectively red and purple) have the same data length, but the green (respectively purple) with a lower sample rate has faster convergence. Note that for each of the data lengths, the figure shows the lowest sampling rates for which training was successful.

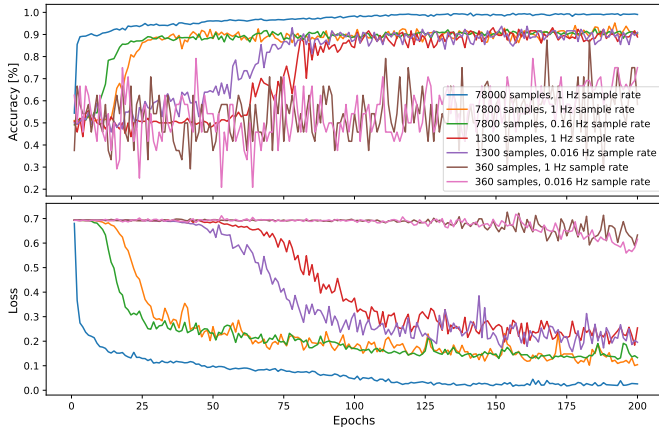


Fig. 9. Resampling evaluation when the number of samples are constant.

Fig. 10 presents zoomed-in data with 1 and 0.16 Hz sample rates. It appears that using short mini-batches with relatively high sample rates causes the oscillations observed in the faulty data to disappear (the data is near-constant over these periods), and that oscillations are important features of the faulty data.

Thus, the reason that the green (purple) result is better than the orange (red) one in Fig. 9 is that the low-resolution faulty mini-batches are easier to classify than high-resolution mini-batches.

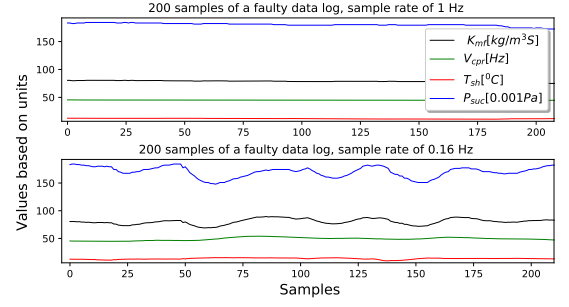


Fig. 10. Two zoomed data logs with different sample rates.

B. Noisy data result

In Subsection IV-B, the method of generating validation data with specified noise is explained. The result of 100 stochastic tests over the CNN model is introduced in Fig. 11. This figure shows how accurately 100 faulty data-sets and 100 non-faulty data-sets are classified. The value at the top of each column shows the distribution of corresponding accuracy values when running 100 stochastic tests. Non-faulty data is classified with higher than 99% accuracy with 100% reliability. Faulty data is classified with better than 97% accuracy in 95 out of the 100 runs.

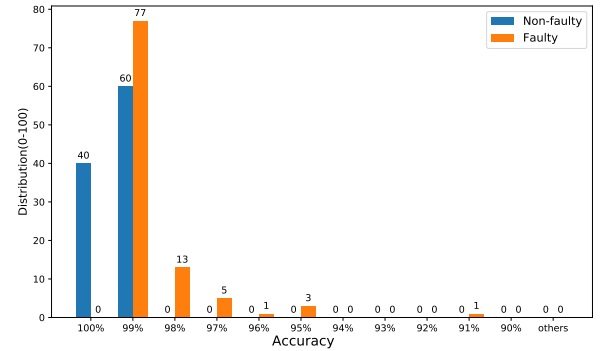


Fig. 11. Distribution of classification accuracy achieved when running the CNN algorithm 100 times using noisy validation data.

C. Result of operating point change

As explained in Subsection IV-C, due to different SRS configurations and loads in the cold room, the data can be varied while the correlation among parameters is preserved. The result of 100 runs is represented in Fig. 12. Note that perturbation of the parameters is limited as explained in Subsection IV-C. The model has good classification capabilities; for the faulty data, 99% accuracy is achieved in all of the 100 runs. On the other hand, non-faulty data are detected correctly with 99% accuracy in 92 out of the 100 runs, while in 8% of the runs non-faulty data was classified with less than 91% accuracy.

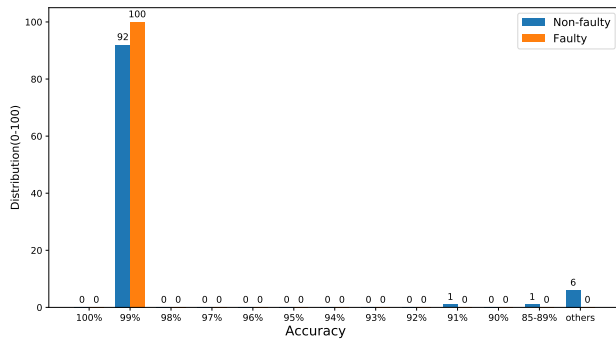


Fig. 12. Distribution of classification accuracy when running the CNN algorithm 100 times using perturbed validation data.

D. False positive analysis

A model is said to give a false positive classification when it incorrectly indicates that a system is faulty while it is, in fact, healthy. False positives must occur as rarely as possible, because it results in unnecessary costs for supermarket owners for changing components or doing inspection. As shown in Fig. 11, 1% false positives were classified in 60 out of the 100 runs using noisy data, while there were no false positives in the remaining 40 runs. Moreover, stochastic perturbation of the data yielded 1% false positives for 92 out of 100 runs, see Fig. 12. This experiment shows more than 9% false positives in 8 out of the 100 runs.

VI. CONCLUSION

In industrial applications, diagnosis of a defective evaporator fan is not always timely, because the inspection is only done when the cooling room temperature exceeds its allowable range. In this paper, a CNN model is applied to detect an evaporator fan fault while the room temperature is actively controlled. Only data from the condensing unit was used because data of the evaporation side is not always available. An evaporator fan fault was emulated on a laboratory SRS, and the data was used to train and analyze the sensitivity of the CNN model to the data quality.

Fast sampling is expensive and monitoring is tedious, therefore one cannot normally expect data of the high quality shown in Fig 6 to be available during normal operation. It was therefore necessary to examine lower sampling rates and shorter data log lengths in order to assess practical classification scenarios.

It was found that using short mini-batches with relatively high sample rates causes the oscillations observed in the faulty data to disappear (the data is near-constant over these periods), and that oscillations are important features of the faulty data.

Moreover, the sensitivity of the model against noisy validation data was studied as well. The noisy and faulty data were classified with better than 98% accuracy for 90 runs out of 100. Maximum 1% false positive classification was achieved when using noisy data.

Validation data acquired at different operating points were classified as well. In these cases, faulty data were classified

with 99% accuracy for all 100 runs. For 92 runs out of 100, only 1% false positive classification was observed, which is a satisfactory result from a practical point of view. It is believed that the higher false positive classification (8% of the runs) can be improved if other random perturbed data is used during the training process. This method can be further developed to classify a number of different faults in SRS systems, allowing automatic early detection of costly faults, which human operators are unlikely to spot during day-to-day operation. Detecting potential faults prevent unnecessary fatigue, leading to lower economic losses to the operator/owner of the system.

ACKNOWLEDGMENT

The authors acknowledge Mads Philipsen, program manager of the Transport department at Bitzer electronics, for insightful technical discussions.

REFERENCES

- [1] J. Arias, *Energy usage in supermarkets - modelling and field measurements*. Stockholm: KTH Industrial engineering and management, 2005.
- [2] C. Ji, Y. Li, W. Qiu, U. Awada, and K. Li, "Big data processing in cloud computing environments," in *2012 12th International Symposium on Pervasive Systems, Algorithms and Networks*, Dec 2012, pp. 17–23.
- [3] A. Krizhevsky and K. A., "Imagenet classification with deep convolutional neural networks," *NIPS*, pp. 1097–1105, 2012.
- [4] J. Z. Y. Huang, "Ga and rbf based real-time fdd for refrigeration units," in *2009 International Symposium on Intelligent Ubiquitous Computing and Education*, May 2009, pp. 22–25.
- [5] Z. Yang, K. B. Rasmussen, A. T. Kieu, and R. Izadi-Zamanabadi, "Fault detection and isolation for a supermarket refrigeration system – part one: Kalman-filter-based methods," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 13 233 – 13 238, 2011, 18th IFAC World Congress.
- [6] Z. Yang, X. Linda, and W. Shengwei, "An intelligent chiller fault detection and diagnosis methodology using bayesian belief network," *Energy and Buildings*, vol. 57, p. 278–288, 02 2013.
- [7] Z. Yang, W. Shengwei, and X. Fu, "Pattern recognition-based chillers fault detection method using Support Vector Data Description (SVDD)," *Applied Energy*, vol. 112, no. C, pp. 1041–1048, 2013.
- [8] K. Assawamartbunlue and M. J. Brandemuehl, "Refrigerant leakage detection and diagnosis for a distributed refrigeration system," *HVAC&R Research*, vol. 12, no. 3, pp. 389–405, 2006.
- [9] Q. Liang, H. Han, X. Cui, H. Qing, and Y. Fan, "Comparative study of probabilistic neural network and back propagation network for fault diagnosis of refrigeration systems," *Science and Technology for the Built Environment*, vol. 24, no. 4, pp. 448–457, 2018.
- [10] M. Najafi, D. M. Auslander, P. L. Bartlett, P. Haves, and M. D. Sohn, "Application of machine learning in the fault diagnostics of air handling units," *Applied Energy*, vol. 96, pp. 347 – 358, 2012, smart Grids.
- [11] S. Li and J. Wen, "Application of pattern matching method for detecting faults in air handling unit system," *Automation in Construction*, vol. 43, pp. 49 – 58, 2014.
- [12] Y. H. Eom, J. W. Yoo, S. B. Hong, and M. S. Kim, "Refrigerant charge fault detection method of air source heat pump system using convolutional neural network for energy saving," *Energy*, vol. 187, p. 115877, 2019.
- [13] J. Chen, "Improved maximum likelihood location estimation accuracy in wireless sensor networks using the cross-entropy method," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, April 2009, pp. 1325–1328.
- [14] J. Yang and G. Yang, "Modified convolutional neural network based on dropout and the stochastic gradient descent optimizer," *Algorithms*, vol. 11, no. 3, 2018.
- [15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, 2015.
- [16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from over-fitting," *Journal of Robotics and Machine Learning*, p. 533, 2014.