Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

A deep learning unsupervised approach for fault diagnosis of household appliances

Francesco Cordoni *, **Gianluca Bacchiega** **, **Giulio Bondani** **, **Robert Radu** ***, **Riccardo Muradore** *

* *University of Verona - Department of Computer Science Strada le Grazie, 15 Verona, 37134-ITALY e-mail: {francescogiuseppe.cordoni,riccardo.muradore}@univr.it*
** *R&D, I.R.S. srl, Padova, Italy e-mail: {bacchiega,bondani}@irsweb.it*
*** *R&D, FirsT srl, Pordenone, Italy e-mail: radu@firstech.it*

Abstract: Fault detection and fault diagnosis are crucial subsystems to be integrated within the control architecture of modern industrial processes to ensure high quality standards. In this paper we present a two-stage unsupervised approach for fault detection and diagnosis in household appliances. In particular a suitable testing procedure has been implemented on a real industrial production line in order to extract the most meaningful features that allow to efficiently classify different types of fault by consecutively exploiting deep autoencoder neural network and *k-means* or *hierarchical clustering* techniques.

Keywords: Fault detection and isolation, Deep Learning, Neural networks, Unsupervised Learning, Autoencoder Neural Networks.

1. INTRODUCTION

Any advanced industrial process needs accurate monitoring in order to detect as soon as possible faults in the machine or even to prevent fault to occur at all. With the advent of modern statistical techniques, led by ongoing development in the field of *machine learning* (ML), the manufacturing process has seen the potential of an efficient *automatic and data-driven* fault detection and diagnosis. Thanks to modern ML techniques, it is now possible to build a strong and reliable automatic fault detection module that can detect faulty machine and out-of-spec products in a production line, identifying an anomalous behaviour and also the nature of the fault. The advantage of implementing such automatic procedure is twofold: on one side it is possible to promptly detect faults with all the benefits coming from an early identification of malfunctioning, on the other side it is possible to considerably reduce misclassified machines with particular reference to undetected faults.

Due to the above mentioned reasons, ML approaches have seen a constant increasing attention in the last ten years as a new paradigm to tackle and solve data-driven and poorly structured problems. The development in sensor technology, computational power and software tools, allowed ML algorithms to be successfully applied to a wide variety of problems with previously unforeseeable accuracy, mainly in action recognition, computer vision, pattern recognition, etc. Recently, ML techniques have also been exploited in the research area of fault detection and isolation, and predictive maintenance for manufacturing processes, see, e.g. Muradore and Fiorini (2011); Susto et al. (2017, 2015); Susto and Beghi (2016).

One of the major disadvantage of ML techniques is that, in order to be able to construct a strong fault classification module, lots of meaningful data have to be collected. Informative features have to be extracted so that models can be trained accurately to detect and isolate faults. This *feature engineering process* is a key aspect to develop a robust and reliable detection system. This involves the correct choice and setting of the most useful sensors. Further, most ML methods require labelled dataset to be trained on, in the sense that for any sample in the dataset a precise indication regarding the state of the system must be provided; such methods are usually referred to in literature as *supervised learning methods*. The above specification is often not available, meaning that indication regarding the state of the system needs to be retrieved having no precise a priori information. Methods that can classify samples without any indication on the working condition of the system are referred to as *unsupervised learning methods*, Kuhn and Johnson (2013), or see, e.g. Routray et al. (2010), Venkataraman et al. (2007) for an application of unsupervised methods to fault detection problems.

We recall that in an unsupervised learning problem, a model detects clusters in which data are divided without any knowledge on the class they belong to. It is clear that supervised methods are stronger and more reliable, nonetheless they necessitate of an expert operator that properly labels each outcome so that a model can be

⋆ This work was supported by the project "PreMANI - Manifattura Predittiva: progettazione, sviluppo e implementazione di soluzioni di Digital Manufacturing per la previsione della Qualità e la Manutenzione Intelligente".

trained. Therefore having a dataset well structured that allows to efficiently use unsupervised methods can avoid the long and costly procedure of suitably labelling each machine (for fault detection) and product (for quality monitoring).

In the present work we propose a fully unsupervised technique for a fault classification problem, where data will be divided into suitable clusters without exact knowledge on fault labels. This technique will be applied to an industrial fault detection problem in a refrigerator test line. Several sensors have been mounted on the devices; in particular temperature, current and power measurements have been collected for any machine in the process. A particular testing procedure, obtained enabling and disabling sequentially some component(s) of the devices, (refrigerator in this study), has been implemented allowing to extract relevant features that characterize the state of the household appliances. Several ML techniques have been developed in order to detect if either the machine is faulty or not, and then subsequently, if a fault occurs, to also detect the exact type of fault. Data from the production line have been collected and each household appliance has been carefully labelled by an expert operator with the type of fault, if any: this labels will be used as ground-truth.

The main contributions of the paper are:

(1) to test different unsupervised techniques to fault diagnosis using real production data;
(2) to present an efficient testing procedure allowing to detect precisely the type and location of the fault on unlabelled datasets.

The paper is structured as follows: in Section 2 the problem and the available data are presented; in Section 3 the proposed methodology is discussed whereas in Section 4 the experimental results are shown. Conclusions are drawn in Section 5 together with the future works.

2. PROBLEM STATEMENT

The objective of the present work is to develop a robust fault classification technique that, monitoring adequately a production line, allows to promptly detect when a fault occurs as well as the exact type of fault. In order to achieve this goal, a precise and highly controlled testing procedure has been designed, implemented and verified on real data. Different types of sensors have been used to collect data from each refrigerator in the production line; in particular *temperature* as well as *current* and *power* have been measured over time. The household appliances under study have two different modes. The testing procedure consists in enabling and disabling different components of the device working alternately in one of the possible modes. Figure 1 shows typical curves of the measured signals. The detailed testing scheme and feature extraction will be treated in later section. The testing procedure, with particular reference to the sequence of different working conditions, allows to extract important information from power, current and temperature profiles, that in turn yields to a set of features characterizing the status of the refrigerator. On these suitable features ML classifiers are designed and trained.

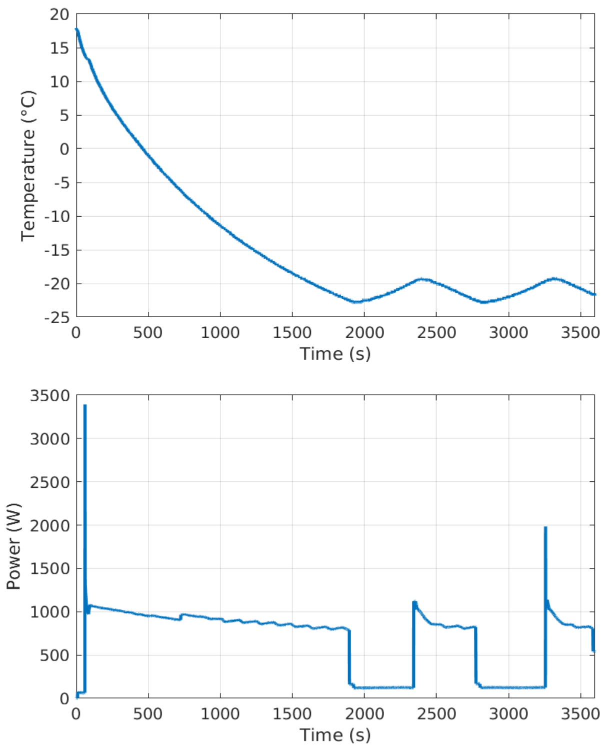

Fig. 1. Temperature and power time series collected during the testing procedure.

2.1 The data

Data consists of 11 features extracted from characteristic power, current and temperature time series alongside with ambient temperature. Each feature is extracted from either power curve or temperature curve and has been chosen to contains useful information for analysing the electromechanical device by expert personal.
The dataset consists of 12,726 machines, of which 12,235 non–faulty. The remaining 491 household appliances have been label as faulty (259 *machine faults* and 232 *server faults*). The dataset contains 2 different types of fault, the former type represents a specific fault of the machine whilst the latter type of fault represents a fault occurred at the software level (server fault). It is worth remarking that the server fault is the most difficult to characterize since the testing procedure has been designed to detect machine faults. This means that this fault is the less important one since it does not require any electrical/mechanical repair. We will therefore consider 3 clusters, where non–faulty devices belong to a cluster, devices with a fault at server level to a second cluster and faults of devices should be collected into the last cluster.

2.2 Feature engineering

As briefly mentioned above, in order to apply ML methods to *fault diagnosis*, a highly controlled testing procedure needs to be implemented. Such a procedure is composed by 6 phases, switching between different operating modes by enabling/disabling particular components of the refrigerator. Table 2 reports the implemented testing procedure; when *mode 1* is on, only the refrigerator works, whereas *mode 2* enables both the refrigerator and freezer; we stress that phases have fixed duration, as reported in Table 2. Some components are further enabled or disabled based on a precise condition that the resulting curve should satisfy.

Component	**Phase 1**	**Phase 2**	**Phase 3**
	Mode 1	**Mode 2**	**Mode 2**
Duration:	**2'**	**15'**	**10'**
Freezer fan	on	off	on
Compressor	on	on	on
Refrigerator valve	on	on	off if $T_1 - T_2 > 3°$
Freezer valve	on	on	on
Resistor	off	off	off
Component	**Phase 4**	**Phase 5**	**Phase 6**
	Mode 2	**Mode 2**	**Mode 2**
Duration:	**25'**	**2'**	**2'**
Freezer fan	off	off	off
Compressor	on	on	off
Refrigerator valve	off	off	off
Freezer valve	on	off if $T_3 - T_4 > 3°$	off
Resistor	off	off	on

Table 2. Testing procedure; T_i means temperature registered at the end of phase i.

At the end of the testing procedure, relevant features from the characteristic power, current and temperature curves are extracted. The extracted features accurately characterize the state of the device and are reported in Table 3.

Feature #	**Meaning**
1	ambience temperature
2	power at the end of phase 4
3	current at the end of phase 4
4	power differential between phase 3 and 4
5	resistance at phase 4
6	power after 16'
7	power after 20'
8	power after 48'
9	current after 16'
10	current after 20'
11	current after 48'

Table 3. Extracted features

Figure 4 shows the existing relation between two of the 11 extracted features. It is evident how some faults (red and blue samples in Figure 4) clearly emerge from non–faulty devices whereas some other types of fault are more complicated to retrieve by simple visual inspection.

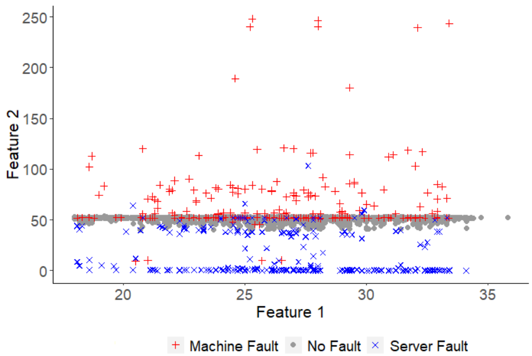

Fig. 4. Scatterplot between Feature 1 and Feature 2

3. FAULT DIAGNOSIS: THE METHODOLOGY

In order to identify faults and malfunctioning in industrial process several techniques may be used. The classical approach is based upon the development of a model, which may be either based on physical laws or identified from input/output measurement data, see, e.g. Qin (2012); Willsky (1976). Although being able to provide strong and accurate fault detection, it relies on the assumption that an accurate mathematical model of the plant is available and that such a model does not change over time (otherwise recurrent identification techniques must be implemented to keep the model updated). Moreover, the identification step often requires a deeply and precise physical/chemical/mechanical knowledge of the process under study. Since this last assumption is rarely true in complex industrial plants, recently *fault identification* methods based on machine learning have proven to provide as accurate results as model-based fault identification. We propose a *two–stage methodology* that combines sequentially different unsupervised methods, where 1) an autoencoder neural network is firstly applied to learn a new compressed data representation, then 2) a clustering technique is applied to the new data representation. This new data representation will be able not only to clearly separate faulty and non–faulty samples, but also to emphasize differences between server faults and machine faults, see, e.g. Chopra and Yadav (2015); Tagawa et al. (2015) for methods exploiting autoencoder neural network in fault detection. It is worth remarking that rather that an extensive automatic feature engineering based on statistical indicators and rolling windows, we extract specific values from, or suitable functions of, the above characteristic curves. This operation has the advantage that our procedure captures the physical sense of the original curves. Therefore, a deep autoencoder is trained to learn a new data representation, so that, based on the new data representation, standard clustering techniques, such as *k–means clustering* and *hierarchical clustering* will be applied to classify data. The proposed approach is compared with both standard approaches, where clustering methods are applied directly to data without pre-processing data through the autoencoder neural network, and clustering methods applied after a dimensionality reduction through *principal component analysis* (PCA).

3.1 Clustering methods

In *unsupervised learning*, unlike *supervised learning*, the target variable is unknown. We are dealing with the unlabelled dataset $\mathcal{D} := \{\mathcal{X}_i\}_{i=1}^N$ composed by N samples, with

$$\mathcal{X}_i := \left(x_1^i, \dots, x_n^i\right) , \quad n \in \mathbb{N}, i = 1, \dots, N ,$$

where $N < \infty$ and $\mathbb{N}$ denotes the natural numbers.
In the *clustering* problem, the main goal is to divide each sample $\mathcal{X}_i$ of the dataset $\mathcal{D}$ into L clusters $S_1, \dots, S_L$; in the present situation the clusters should correspond either to the label *fault/no fault* or to different types of fault.
There exists several *clustering methods*; one of the most popular is known as *k-means clustering* and it is a *centroid-based clustering method*. In this type of method each sample is assigned to the cluster S_k such that within each cluster the average dissimilarity of the observations from the cluster mean is minimized, see, e.g. Friedman et al. (2001). In particular, one aims at finding the cluster S_k that minimizes the following loss function

$$\sum_{k=1}^{L} \sum_{x^i \in S_k} \| \left(x_1^i, \ldots, x_n^i\right) - \bar{x}_k \|^2 ,$$

being $\bar{x}_k$ the $k-$th cluster mean.
Another different type of *clustering method* is the *hierarchical clustering*, see, e.g. Friedman et al. (2001). As *k-means* clustering is based on the assumption of L clusters specified by the user, *hierarchical clustering* on the contrary does not require any prior specification on the number of clusters but the best number is searched based on a measure of dissimilarity between groups of observations specified by the user. The output is a hierarchical representations of the dataset, allowing to derive a hierarchy in the clusters. Several dissimilarity measures can be used, such as the standard Euclidean norm or squared Euclidean norm.

3.2 The Autoencoder Neural Network

The AutoEncoder (AE) neural network, see, e.g. Vincent et al. (2010), is an *unsupervised* network used to learn a new representation for a given dataset, typically with the final goal of reducing dimensionality or removing noise. Main applications of autoencoders are found in highly unbalanced datasets where the majority of data belongs to a given class. The classification problem under analysis in the present work is of this type since most of the data belong to the no–fault class. The autoencoder consists of two parts: the *encoder* and the *decoder*. An AE neural network works similarly to a classical feed–forward network, the first layer receives the raw input data $\mathcal{X}_i$ and compresses them into a lower dimensional representation $\mathcal{H}_i$ through a non–linear function σ^1,

$$\mathcal{H}_i = \sigma^1 \left(W^1 \mathcal{X}_i + b^1\right) ,$$

whereas the decoder part takes the new representation $\mathcal{H}_i$ as input and then reconstructs the input with a non–linear function

$$\mathcal{Z}_i = \sigma^2 \left(W^2 \mathcal{H}_i + b^2\right) ,$$

where σ^1 and σ^2 are suitable activation functions, W^1 and W^2 are weights matrices, and b^1 and b^2 are biases.
A further typical structure of the autoencoder is the bottleneck joining the *encoder* and the *decoder*, see, e.g. Bengio et al. (2009); LeCun et al. (2015).
In particular, given the unlabelled dataset $\mathcal{D}$, the AE solves the following optimization problem

$$\min_{W,b} \frac{1}{2N} \sum_{i=1}^{N} \left\| \mathcal{X}_i - \sigma^2 \left(W^2 \sigma^1 \left(W^1 \mathcal{X}_i + b^1\right) + b^2\right) \right\|^2 + \\ + \frac{\lambda}{2} \left(\left\| W^1 \right\|^2 + \left\| W^2 \right\|^2 \right) ,$$

where λ is a constant weight decay parameter needed for regularization reasons.
Given $\mathcal{D}$, the final goal of the autoencoder is to learn the function f that maps $\mathcal{X}_i \in \mathbb{R}^n$ into a lower dimensional vector $\mathcal{H}_i \in \mathbb{R}^d$, $d < n$, that is

$$\mathcal{H}_i = f\left(\mathcal{X}_i\right) .$$

3.3 Evaluation of results

The *confusion matrix* reported Table 5 will be considered to easily visualize the results of a classification method. In particular in a confusion matrix, predicted classes (rows) are reported against actual classes (columns) in order to highlight true prediction but also wrong prediction, Kuhn and Johnson (2013).

	Predicted class		
Actual class		**Fault**	**No Fault**
	Fault	True Positive (TP)	False Positive (FP)
	No Fault	False Negative (FN)	True Negative (TN)

Table 5. Confusion matrix

Due to the higher portion of non faulty devices in our dataset [1], we will not base our analysis on accuracy, which is the proportion between right predictions and total number of samples; instead we will focus on other metrics such as *positive predicted value* (PPV), *negative predicted value* (NPV), *true positive rate* (TPR) and *true negative rate* (TNR) which provide measures of how frequently a faulty device is classified as non–faulty, and vice-versa. In particular given the nature of our problem, our main focus will be on PPV and TPR: metrics that emphasize how many faults have been misclassified. Table 6 reports the definitions of those metrics, Kuhn and Johnson (2013).

Metric	**Definition**
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
Positive predicted value (PPV)	$\frac{TP}{TP+FP}$
Negative predicted value (NPV)	$\frac{TN}{TN+FN}$
True positive rate (TPR)	$\frac{TP}{TP+FN}$
True negative rate (TNR)	$\frac{TN}{TN+FP}$

Table 6. Metric definitions

4. FAULT DIAGNOSIS: THE CASE STUDY

The *unsupervised methods* introduced in Section 3 will be tested on data from a real production line of household appliances.
We will test six methods. First we will apply *k-means clustering*, second we will apply the same *k-means* method to a new representation of the data learned via *principal component analysis* (PCA) and, third, *k-means clustering* is applied after an *autoencoder neural network* is used to learn a new data representation. Finally, *hierarchical clustering* will be applied to raw data, PCA representation and *autoencoder* representation. It is worth mentioning that both PCA and the autoencoder are unsupervised methods so that the procedure is in fact fully unsupervised. We will require the method to divide the dataset into three clusters, that in principle should coincide with *server fault*, *machine fault* and *no fault*.
The confusion matrices in Tables 7 summarize the results applying *k-means*, *PCA* plus *k-means* and the autoencoder plus *k-means* clusterings; it can be seen that all methods perform poorly. In particular, although the autoencoder representation, as it can be seen in Figure 10, highlights three clusters, *k-means* method fails to understand real

[1] Such a small number of faults is due to the fact that the production line under study has been accurately optimized in the last years for mass production.

division between clusters, with very poor results. We stress that, given poor results showed by the *k-means* method, we will not report graphical results for this method but only confusion matrices.

		Predicted class		
		Machine Fault	**Server Fault**	**No Fault**
Actual class	**Machine Fault**	222/184/221	10/0/7	0/48/4
	Server Fault	182/74/182	52/156/50	25/29/28
	No Fault	0/6033/0	5340/0/5956	6895/6202/6279

Table 7. Confusion matrix for *k-means/PCA+k-means/autoencoder+k-means*.

In particular, the poor results emerging after the application of the autoencoder might be motivated by the fact that clusters, as defined above, are hierarchical, in the sense that the two faulty clusters, i.e. *machine fault* and *server fault*, belong to the same cluster of faulty devices. We report that similar results hold even if one tries to add more clusters to the method.
Based on this intuition we have therefore applied *hierarchical clustering*, so that we are trying to divide data into three hierarchical clusters. In particular the dissimilarity measure highlights a structure composed by two main clusters, which should correspond to *no fault* and *fault*, this last cluster further divided into two more clusters, *machine fault* and *server fault*.
The confusion matrix in Table 11, report the results for the hierarchical clustering applied directly to raw data and *PCA* plus *hierarchical clustering*; Figure 8 reports results *PCA* plus *hierarchical clustering*; in particular top panel represents real data classes whereas the bottom panel represents data divided into cluster using the proposed method. As for the *k-means clustering* case, also applying *hierarchical clustering* directly to raw data or after PCA has been applied, shows poor results. In particular, the *hierarchical clustering* exhibits a typical problem of highly unbalanced dataset, where the majority of samples are classified as non–faulthy, resulting in high accuracy but very low true negative rates. On the contrary PCA seems to separates three clusters, with some of the server faults being isolated from the rest, but still no clear clusters emerge from the new representation. This might be due to some non–linear relations in the data, that can be learned via a deep autoencoder neural network.

		Predicted class		
		Machine Fault	**Server Fault**	**No Fault**
Actual class	**Machine Fault**	0/14	1/0	258/218
	Server Fault	8/1	8/156	216/102
	No Fault	0/0	1/	12234/12234

Table 9. Confusion matrix for *hierarchical clustering/PCA+hierarchical clustering*

Figure 10 and the confusion matrix in Table 11, report the results for the hierarchical clustering applied after the *autoencoder neural network* has been used to learn the new data representation. It can be seen how now three clusters clearly emerges in the new representation space, so that *hierarchical clustering* is able to detect a great proportion of data classes with high accuracy, with only 113 misclassified devices out of 12726.

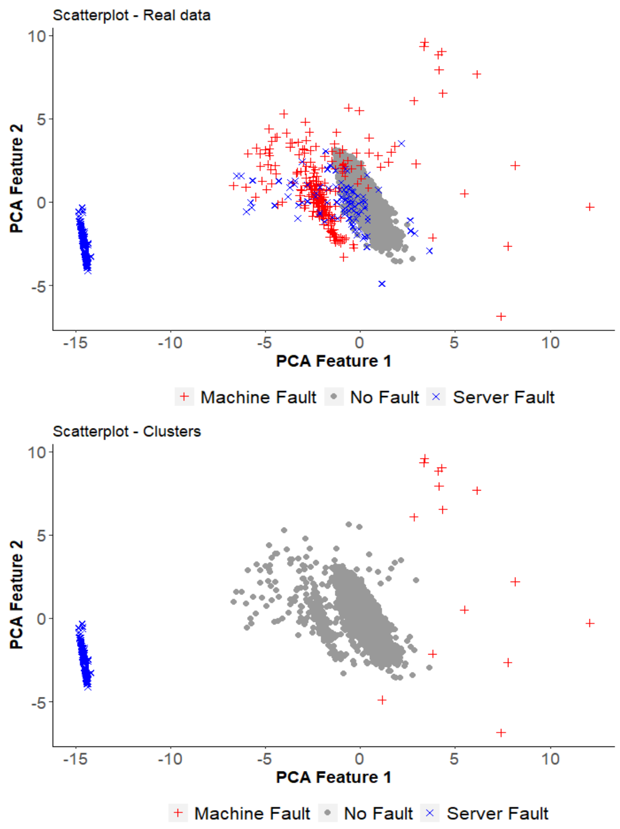

Fig. 8. Representation of real data (top panel) and division into clusters (bottom panel) for *PCA* plus *hierarchical clustering*

		Predicted class		
		Machine Fault	**Server Fault**	**No Fault**
Actual class	**Machine Fault**	222	10	0
	Server Fault	26	145	88
	No Fault	0	0	12235

Table 11. Confusion matrix for *autoencoder+hierarchical clustering*

Table 12 reports main metrics with respect to the two classes Fault and No Fault. It appears that the autoencoder with hierarchical clustering has better performance in terms of accuracy, NPV and TNR metrics. Regarding PPV and TNR metrics, k-means algorithm shows better performance. It is worth highlighting that, although k-means shows excellent PPV and TNR performance, the results for the other metrics are very poor. This means that, as reported in Table 7, k-means method has a large number of false negative, predicting that a sample presents some type of fault even though the machinery is working properly. This fact goes against the requirement of a fully automated and reliable testing procedure where the operator has to intervene only when the algorithm reports a (real) malfunctioning. A TPR of 8% means that only the 8% of total faulty predicted samples presents a real fault. The autoencoder with hierarchical clustering is then the more reliable solution since the new representation learned via the autoencoder allows to efficiently separate samples according to the type of fault.

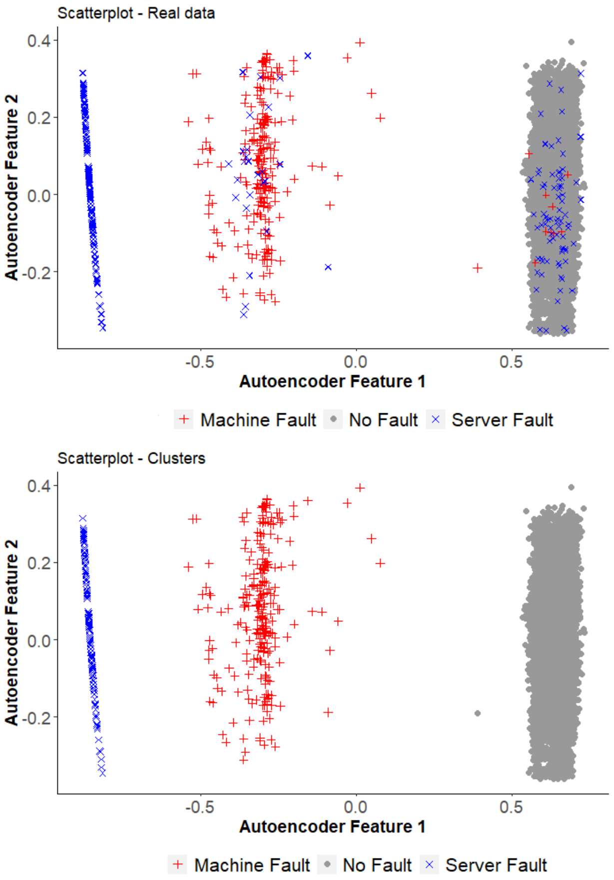

Fig. 10. Representation of real data (top panel) and division into clusters (bottom panel) for *autoencoder* plus *hierarchical clustering*

Method	Accuracy	PPV	NPV	TPR	TNR
k-means	57.8%	**94.9%**	56.3%	8%	**99.6%**
Hierarchical	96.2.8%	34.6%	99.9%	94.4%	96.2%
PCA + k-means	52%	84.3%	50.6%	6%	98.7%
PCA + Hierarchical	97.4%	34.8%	99.9%	99.4%	97.4%
AE + k-means	52.9%	51.3%	7.1%	100%	99.4%
AE + Hierarchical	**99.3%**	82%	**100%**	**100%**	99.2%

Table 12. Results for clustering methods

5. CONCLUSION

This work shows how a fully unsupervised two–stage method that combines modern autoencoder neural network and classical clustering methods can be successfully applied to the *fault detection* and *diagnosis* problems. Proper feature extraction based on precise physical meaning has been collected on power, current and temperature curves to obtain relevant features to base the ML methods upon. We have shown that the dimensionality reduction capability of an *autoencoder neural network* can improve performance of standard clustering method for highly unbalanced datasets. More important we have shown how the autoencoder can also learn a new data representation that can lead to high accuracy, especially in the fault diagnosis case. The results reported in Table 12 show that the proposed algorithm presents the higher accuracy with respect to the most important metrics. This is clearly important on real manufacturing processes where a fully unsupervised testing procedure allows to ask for an expert human intervention only when really needed. Future possible extensions to the above method include the usage of autoencoder directly on raw time series data, such as the *long–short time memory* (LSTM) autoencoder. This allows to avoid feature extraction, exploiting feature extraction capability typical of deep learning methods.

REFERENCES

Bengio, Y. et al. (2009). Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1), 1–127.

Chopra, P. and Yadav, S.K. (2015). Fault detection and classification by unsupervised feature extraction and dimensionality reduction. *Complex & Intelligent Systems*, 1(1-4), 25–33.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Kuhn, M. and Johnson, K. (2013). *Applied predictive modeling*, volume 26. Springer.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. nature 521 (7553): 436. *Google Scholar*.

Muradore, R. and Fiorini, P. (2011). A pls-based statistical approach for fault detection and isolation of robotic manipulators. *IEEE Transactions on Industrial Electronics*, 59(8), 3167–3175.

Qin, S.J. (2012). Survey on data-driven industrial process monitoring and diagnosis. *Annual reviews in control*, 36(2), 220–234.

Routray, A., Rajaguru, A., and Singh, S. (2010). Data reduction and clustering techniques for fault detection and diagnosis in automotives. In *2010 IEEE International Conference on Automation Science and Engineering*, 326–331.

Susto, G.A. and Beghi, A. (2016). Dealing with time-series data in predictive maintenance problems. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–4.

Susto, G.A., Schirru, A., Pampuri, S., McLoone, S., and Beghi, A. (2015). Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics*, 11(3), 812–820.

Susto, G.A., Terzi, M., and Beghi, A. (2017). Anomaly detection approaches for semiconductor manufacturing. *Procedia Manufacturing*, 11, 2018–2024.

Tagawa, T., Tadokoro, Y., and Yairi, T. (2015). Structured denoising autoencoder for fault detection and analysis. In *Asian Conference on Machine Learning*, 96–111.

Venkataraman, G., Emmanuel, S., and Thambipillai, S. (2007). A cluster-based approach to fault detection and recovery in wireless sensor networks. In *2007 4th International Symposium on Wireless Communication Systems*, 35–39.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec), 3371–3408.

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