2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)
August 23-27, 2021. Lyon, France

A Comparative Evaluation of Deep Learning Anomaly Detection Techniques on Semiconductor Multivariate Time Series Data

Philip Tchatchoua, Guillaume Graton, Mustapha Ouladsine and Michel Juge

2021 IEEE 17th International Conference on Automation Science and Engineering (CASE) | 978-1-6654-1873-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/CASE49439.2021.9551541

Abstract— **In industrial processes, keeping equipment units in good operating conditions while reducing maintenance costs is one of the most important objectives to improve productivity. One of the ways to do so is to early detect equipment dysfunction. This can be done by analyzing the massive amounts of data collected via numerous sensors during production activities. Thanks to the advantages of distributed architecture and computation efficiency improvements, deep learning methods have gained much interest and have been investigated by researchers for thorough industrial data analysis, notably for anomaly detection. A comparative evaluation of data-driven deep learning methods used to detect anomalies occurring during equipment processing is proposed. This evaluation is done on a total of six methods, their ability to detect anomalies on raw sensor data, collected on semiconductor machines, with variable correlations and temporal dependencies is discussed. The evaluation gives an insight on the industrial performances on the methods and shows how the supervised learning methods outperform the other models with less training time but need labelled data meanwhile some self-supervised learning methods have good detection performances with training done on normal data only.**

Index Terms— **Anomaly detection, data-driven methods, deep learning, multivariate analysis, raw sensor data, semiconductor manufacturing.**

I. Introduction

The semiconductor industry is a highly complex and competitive one, implying the manufacturing of different types of products in the requested quantities, with the required quality, at the lowest cost while respecting deadlines. Industry 4.0 is in perfect harmony with these goals. Its concept emphasizes, among other things, on a continuous collection of physical and contextual parameters of production equipment [1]. On each production equipment, diverse sensors measure the environment in which the silicon wafers are processed and sensor data like temperatures, intensities, frequencies, pressures, gas flow rates, consumed power are collected, generally every second. This multi-sensor data is used by Advanced Process Control (APC) for equipment monitoring.

APC is an umbrella term that refers to different types of process control tools, often used to solve multivariable control problems. The goal of APC is to increase overall equipment effectiveness, to optimize the process by minimizing manufacturing defects. It is therefore important to define optimized analysis methods capable of determining whether a variable is at the expected level or if, on the contrary, the variable appears to be off-standards or drifting. This is the goal of Fault Detection and Classification (FDC), one of the components of APC.

The complexity of the fault detection is twofold: the volume of data collected every second and the heterogeneity of the data. About data heterogeneity, a variable can be very stationary, have a profile according to a normal distribution or on the contrary very fluctuating with random peaks. A variable can be numerical: temperature, pressures, flow, power, but also contextual: equipment, product, recipe, etc. The data collected is "raw" and thus not processed. It is generally segmented and reduced into summary statistics data [2] like the mean, the minimum, the maximum, the standard deviation, skewness, kurtosis or percentiles, which is then used for FDC. This reduction involves a risk. Indeed, a set of data cannot always be characterized by its reduced statistical representation [3]. It is therefore necessary to process, sort and analyze this raw data as soon as it is collected using advanced techniques.

The equipment raw data exhibit significant temporal dependencies and can be seen as multivariate time series from multiple correlated sensor signals. The systems studied being mostly dynamic, the time series collected show some difference in the number of observations and values from one wafer to another, which introduces data desynchronization. FDC aims at identifying anomalies through data mining and analysis. With the high volumes and dimensionality of such data, traditional knowledge-based and machine learning approaches become very difficult to implement. The breakthroughs in graphical processing units (GPU), wireless networks, cloud computing and deep learning provide consistent material for the emergence of next generation FDC systems [4][5].

The remainder of this paper is organized as follows: Section II presents the industrial context. Section III describes the evaluated deep learning models. Section IV presents a performance evaluation experiment on a simulated semiconductor multivariate dataset and discusses the results obtained. Finally, Section V concludes the paper and discusses future research topics.

II. Industrial Context

Semiconductor manufacturing pursues a High Mix Low Volume activity, characterized by a wide variety of technologies, low production volumes, and short life products, involving peculiar processes such as batch

*Research supported by STMicroelectronics company and MadeIn4 European project under grant agreement No. 826589.

P. Tchatchoua, G. Graton and M. Ouladsine are with Aix Marseille Univ, Université de Toulon, CNRS, LIS (UMR 7020), F-13397 Marseille Cedex 20, France (e-mail: philip.tchatchoua@lis-lab.fr).

G. Graton is with Ecole Centrale Marseille, Technopôle de Château-Gombert, F-13451 Marseille Cedex 13, France.

M. Juge is with STMicroelectronics company, ZI – Rousset-Peynier, F-13106 Rousset, France (e-mail: michel.juge@st.com).

manufacturing, or reentrant lines. There exist a number of techniques for FDC.

Model-based FDC approaches rely on a deep knowledge and understanding of the structure and behavior of the system and its components, as well as the physical processes involved. In the case of particularly complex systems or processes, obtaining an accurate model can be difficult or even impossible. This is particularly the case in the semiconductor industry, where production equipments are complex entities made up of a large number of subsystems, processes, and operating modes. The great complexity of the physical processes involved [26], the impact of environmental parameters as well as the properties of the materials also limit the availability of relevant structural or functional models. In addition, several factors limit the relevance and reliability of expert knowledge: frequent process variations motivated by the arrival of new products; reconfigurations occurring during the production process; relative recency of production equipment induced by the continuous improvement of manufacturing techniques. The consequences are a limited amount of reliability data, and hardly any physical or mathematical models, these being extremely expensive to develop and quickly becoming obsolete. This constitutes a major obstacle to the use of any model-based FDC approach.

In a typical semiconductor fab, hundreds of diverse equipment process silicon wafers in a multistep manufacturing scheme, to produce various electronic chips. Being a highly sensitive manufacturing process, requiring extreme precision (on a nanometric scale), numerous sensors monitor every production step on each equipment to ensure the required product quality. This leads to a huge volume of data collected every second, for many variables (roughly 60 per equipment in general), which has to be processed with the most appropriate methods to extract the best information, critical for process control of which FDC is an important component.

Adding to the high data volumes, the technical challenges of industrial FDC are the high complexity of sensor signals, the distortion in data in the form of time/value warping and the unbalanced samples of faulty events. Statistical analysis coupled to machine learning methods have been historically set up for FDC. The machine learning methods can be divided into unsupervised, self-supervised and supervised learning. Self-supervised and unsupervised learning is widely used for FDC in the semiconductor industry since fault wafers are rare. The desynchronization of data in the form time and/or value warping justifies the importance of extensive data preprocessing before usage. Preprocessing methods like Dynamic Time Warping (DTW) makes it possible to transform the raw measurements collected into data usable by Principal Component Analysis (PCA) [6]-[9] and One-class Support Vector Machines (OC-SVM) [10]. PCA and OC-SVM are used as self-supervised learning methods for outlier detection. The Exponentially Weighted Moving Average (EWMA) filter makes it possible to consider the dynamics of the system during the performance of an operation [7]. Coupled to an EWMA filter, multiway PCA is used for dimensionality reduction thereby doing a crucial selection of variables used by Hotelling's T^2 and squared prediction error charts. PCA is also used to extract features for a k-Nearest Neighbors (k-NN) classification of normal and faulty wafers [11].

Anomaly detection is the process of explicitly identifying data outliers and abnormalities also referred to as anomalies. Anomalies are data points which differ significantly from the previous normal data points and are rare in the dataset. The aforementioned FDC methods mostly do anomaly detection applied to multivariate time series data in a self- and unsupervised way. All these methods have shown some relevance on small datasets [12] but show limitations on large datasets. With the great number of equipment and variables to monitor, managing the subsequent number of models resulting from the industrial implementation of these methods is bold task. Furthermore, regarding current data volumes, the performances of the models in preprocessing and especially data modeling are not optimal in a big data environment. The big data paradigm creates a need for anomaly detection methods able to yield great industrial performances on large data volumes. With the enhancement of deep learning coupled to the breakthroughs in cloud computing, and GPUs, the techniques employed to endeavor anomaly detection have hence evolved towards deep learning methods [13].

III. Deep learning Methods for anomaly detection

In this section, supervised and self-supervised anomaly detection methods for industrial multivariate data, based on three deep learning models are briefly introduced. For supervised anomaly detection, the model is fitted with labeled training data and performances evaluated with labeled test data, which strongly reflects typical pattern recognition classification. For self-supervised anomaly detection, the model is fitted with normal data points, without anomalies in order to learn the normal behavior and anomalies are detected during evaluation as deviations from the normal class. Where supervised anomaly detection does its training with two classes or more classes, normal and faulty, self-supervised anomaly detection does its training only with one class, representing the normal data [27]. Depending on the methods employed, the data can be in the form of features (Isolation Forest, OC-SVM) or raw data (PCA, OC-SVM, neural networks).

A. CNN based methods

Among currently available deep learning models, Convolutional Neural Networks (CNNs) have the center place in the field of image recognition due to their high classification performances. CNNs [14] are a type of feedforward neural networks dedicated to the processing of grid-shaped data. A CNN is composed of two parts.

A first part which does feature extraction and has an input layer, convolutional layers, and pooling layers. The convolutional and pooling layers, which are stacked in the network, serve as feature extractors. The convolutional layers have convolutional kernels which go through the data and craft features while the pooling layers do feature selection. Some other layers such as batch normalization layers are also used to craft robust features.

The second part does a classification task using the features extracted by the first part and has fully connected

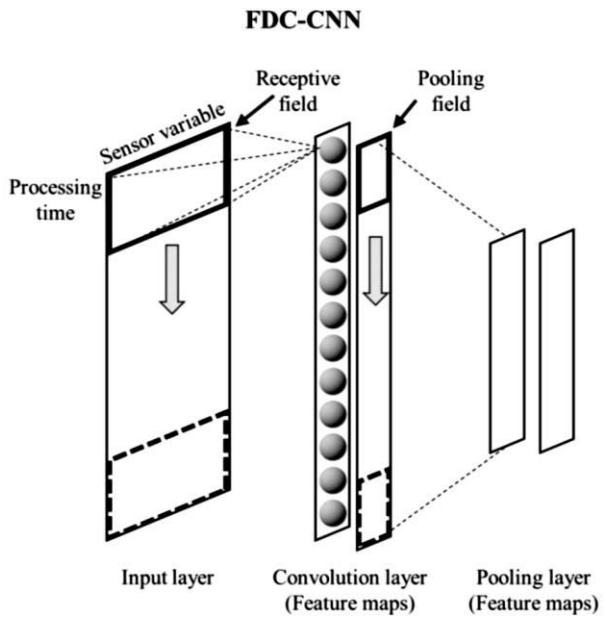

Figure 1: Feature extraction in FDC-CNN

layers and an output layer. The features are generally received from the last pooling layer and serve as input for the fully connected layers which together form a multilayer perceptron (MLP).

Nowadays, CNNs are highly used for process monitoring in various manufacturing systems due to their capacity to automatically extract features from raw data. CNNs are a feasible choice for dealing with multivariate time series since they are represented as matrices. In the semiconductor manufacturing industry, Lee et al. [15] proposed FDC-CNN, a supervised anomaly detection method which achieved high classification performance in fault detection on a Chemical Vapor Deposition (CVD) process dataset consisting of multivariate time series. Fig. 1 illustrates the proposed architecture. FDC-CNN detects and captures inter-variable correlations during feature extraction by sweeping the time axis of the two-dimensional input with convolutional kernels. Considering an input x of size $T \times K$, T being the length of the time series and K the number of variables in the dataset and a non-squared kernel receptive field of size $F \times K$, F being the length of the convolutive kernel, the output of a node of the feature map is given as:

$$y(c) = g\left(\sum_{i=1}^{F}\sum_{j=1}^{K} x(f_i, j)w(i,j) + b\right), \qquad (1)$$

where $f_i = cS + i$ with $0 \le c \le (T-F)/S$ the number of extracted feature map vectors by a convolutional layer, $w \in \mathbb{R}^{F\times K}$ is the kernel weight, b the bias, S the stride length and g the activation function, which can be the ReLU, the sigmoid or the tanh activation. Feature dimension reduction is done by the pooling layers. Kim et al. [16] proposed a modified version of FDC-CNN (afterwards referred to as Attention FDC-CNN) combining a self-attention mechanism with a CNN and tested the method on an etch process dataset. The self-attention mechanism [17] is designed to ignore the irrelevant parts and focus on the relevant parts of a sequence by attributing attention weights via a probability distribution. The attention weights are learned by backpropagation and have the advantage of giving insight to fault diagnosis.

B. LSTM based methods

Recurrent Neural Networks (RNNs) are dedicated to sequence processing of which Long Short-Term Memory (LSTM) [18] is the most popular architecture. RNNs perform the same task for every element of a sequence, caching information from computations on previous elements of the sequence through loops. LSTM cells have four hidden layers which interact with each other in a very peculiar way in order to retain information and overcome the vanishing gradient problem experienced by RNNs. The retained information also called memory is passed from one cell to another. The interactions of the memory with new inputs and cell outputs are regulated by employing three multiplicative gates, resulting from a well-designed network of activation functions. The forget, input and output gates leverage LSTMs ability to learn long term dependencies in sequences.

Time series inherently being sequences, LSTMs are capable of accurately modelling both temporal and multivariate correlations. Malhotra et al. [19] proposed LSTM-AD, a self-supervised anomaly detection method based on stacked LSTMs. The method is fundamentally different in that it does not directly produce anomaly scores for the analyzed time series. LSTM-AD is used as a predictor to model the behavior of normal time series. The probability distribution of prediction errors is used to calculate anomaly scores and define a threshold to detect anomalies. During the training phase, the model is trained to minimize the prediction errors computed for each time step of the series as $e^{(i)} = \left\|x^{(i)} - \hat{x}^{(i)}\right\|$, where $x^{(i)}$ and $\hat{x}^{(i)}$ are the targeted and predicted values respectively and $i \in \{1, 2, \ldots, T\}$. Malhotra et al. [20] later proposed EncDec-AD, an LSTM-based Encoder-Decoder scheme for anomaly detection on multi-sensor time series. Fig. 2 illustrates the proposed architecture. The encoder learns a fixed-size vector representation of input time series and the decoder reconstructs the input from the representation. The encoder-decoder is trained to reconstruct the timeseries in the reverse order and the reconstruction error is used to compute an anomaly score for the time series such that the higher scores indicate a higher anomaly likelihood. Considering the hidden state of the encoder (resp. decoder) at the time step i as $h_E^{(i)} \in \mathbb{R}^c$ (resp. $h_D^{(i)} \in \mathbb{R}^c$), c is the number of LSTM cells, the reconstructed values are given by (3) as seen below:

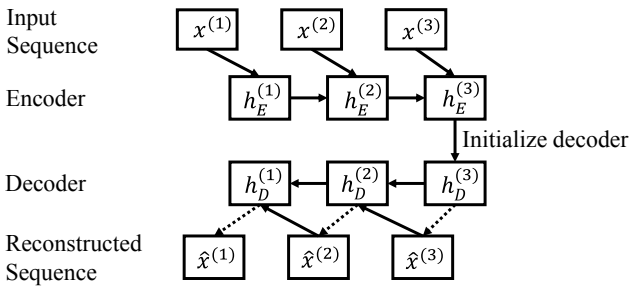

Figure 2: LSTM-based Encoder-Decoder inference for 3 time steps

$$h_E^{(i)} = W_1 x^{(i)} + b_1, \quad (2)$$

$$\hat{x}^{(i)} = W_2 h_D^{(i)} + b_2, \quad (3)$$

where $h_D^{(T)} = h_E^{(T)}$ is the final state of the encoder and thus, the initial state of the decoder. The anomaly scores for any point $x^{(i)}$ the reconstructed values are calculated as follows :

$$a^{(i)} = \left(e^{(i)} - \mu\right)^T \Sigma^{-1} \left(e^{(i)} - \mu\right), \quad (4)$$

where μ and Σ are the parameters of a multivariate gaussian distribution $\mathcal{N}(\mu, \Sigma)$.

With these anomaly scores, a threshold to set a decision boundary is calculated using the mean and the standard deviation of the anomaly scores.

$$\tau = \mu + 2\sigma \quad (5)$$

C. Autoencoder based methods

Autoencoders (AEs) are simple feed-forward neural networks with a bottleneck structure widely used as unsupervised feature learning algorithms [21]. AEs approximate the identity function by reconstructing the input at the output. AEs are particularly useful to extract nonlinear features from various types of data and are thus used for multivariate time series. AEs have two parts: an encoder which converts high-dimensional input data into low-dimensional features and a decoder that reconstructs the input from the low-dimensional features. Considering an input $x \in \mathbb{R}^{T \times K}$, the encoder maps the input to a code $c \in \mathbb{R}^{P \times K}, P < T$ and the decoder maps de code to the reconstructed output $\hat{x} \in \mathbb{R}^{T \times K}$ as follows:

$$c = g_1(W_1 x + b_1), \quad (6)$$

$$\hat{x} = g_2(W_2 c + b_2), \quad (7)$$

where g_1, g_2 are activation functions and W_1, W_2, b_1, b_2 are weights and biases learned by backpropagation. Fig. 3 illustrates the described architecture. To minimize the

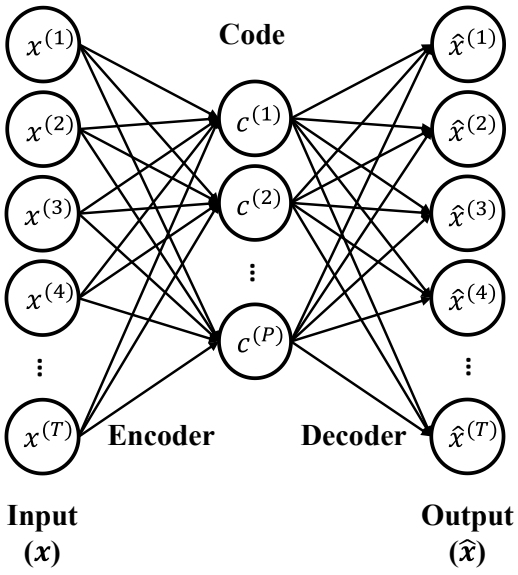

Figure 3: Stacked Autoencoder

reconstruction error which represents the difference between x and $\hat{x}$, a cost function is introduced to measure this difference on an entire dataset. It has two terms: a squared error term and a weight decay term for weight regularization in order to address overfitting. For the training of N data samples, the cost function is:

$$J = \frac{1}{2N} \sum_{t=1}^{N} \left\| x^{(t)} - \hat{x}^{(t)} \right\|^2 + \frac{\lambda}{2} \sum_{i=1}^{2} \|W_i\|^2, \quad (8)$$

where t is the t[th] training sample and λ the coefficient of the regularization term which determines the balance between training error and generalization ability. The reconstruction error of the neural network is used to compute the root-mean-square error which is used as an anomaly score. For a well-trained AE, the reconstruction error is low for normal data instances and is relatively high for data instances presenting anomalies. A decision threshold can be set and optimized for anomaly detection. EncDec-AD can be seen as an LSTM based autoencoder method for anomaly detection.

The denoising autoencoder (DAE) is a variant of AEs which can learn features that are more robust to input noise and thus very useful for real-world signals. For DAEs, the input data is corrupted with additive Gaussian noise and the network is trained to reconstruct the uncorrupted data. Chen et al. [22] proposed a self-supervised FDC method based on convolutional sparse autoencoders (CSAE-AD) and the corresponding denoising (CDSAE-AD). Convolutional sparse autoencoders differ from simple autoencoders with the usage of convolutional kernels and the introduction in the cost function of a sparsity penalty [28], based on the Kullback-Leibler divergence [29]. The new cost function is given as:

$$J = \frac{1}{2N} \sum_{t=1}^{N} \left\| x^{(t)} - \hat{x}^{(t)} \right\|^2 + \frac{\lambda}{2} \sum_{i=1}^{2} \|W_i\|^2 + \beta \sum_{j=1}^{P} D(\rho || \tilde{\rho}_j), \quad (9)$$

where β controls the weight sparsity term and $D(\rho||\tilde{\rho}_j)$ is Kullback-Leibler divergence between ρ and $\tilde{\rho}_j$.

In this paper, the anomaly scores and the decision boundary threshold for CSAE-AD and CDSAE-AD are the calculated with the same method as in (4) and (5).

Later, Lee et al. [23] used stacked denoising autoencoders to propose an FDC model for simultaneous extraction of noise-tolerant features and their classification, in the semiconductor industry.

IV. Experiment

A. Data and Setup environment

The data is collected every second, for a recipe lasting approximately 150 seconds, from one processing module of a unique equipment. A total of 11 variables are monitored, which are simulations with a nature very close to real production variables like gas flows, pressures, temperatures, capacitance of an etch tool. Strong intervariable correlations are noticed for some variables (e.g. 1 & 4, 2 & 11). A total of 5000 normal data samples and 2000 faulty data samples constitute the dataset for this study which gives a ratio of 28.6% of faulty data, which is a good one with regards to the rarity of faulty data in the semiconductor industry. During

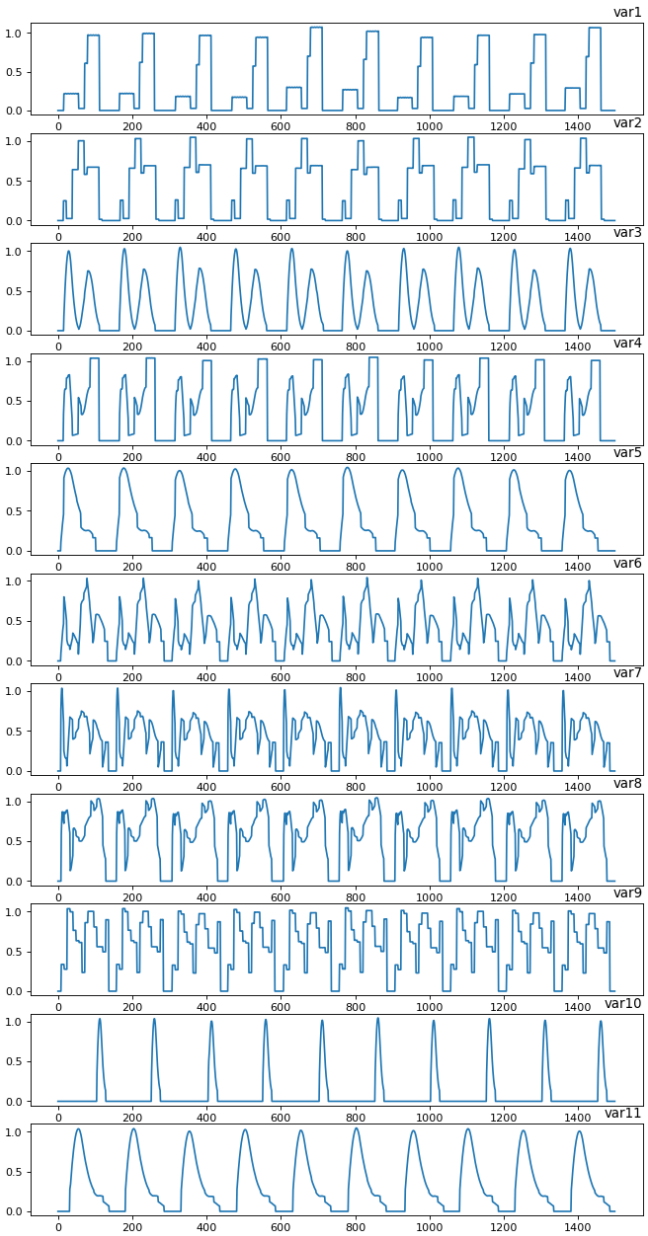

Figure 5: Normalized raw sensor time series

preprocessing, the data is normalized to range between 0 and 1. Fig. 4 illustrates 10 data samples without faults.

The time-varying features of multivariate time series are critical to identify faults and the distinction between various types of faults is a challenging task due to the complex correlations among variables. The focus here is to detect faults, whatever the type. The classification of the various detected faults is not investigated by the current comparative study. Nonetheless, the evaluation dataset presents 5 different types of faults, equally distributed in the dataset (400 samples per fault type) on which the evaluation will be done individually.

The 5 fault types in the data represent common faults occurring during wafer processing. They occur on various variables and are either atomic or aggregate anomalies.

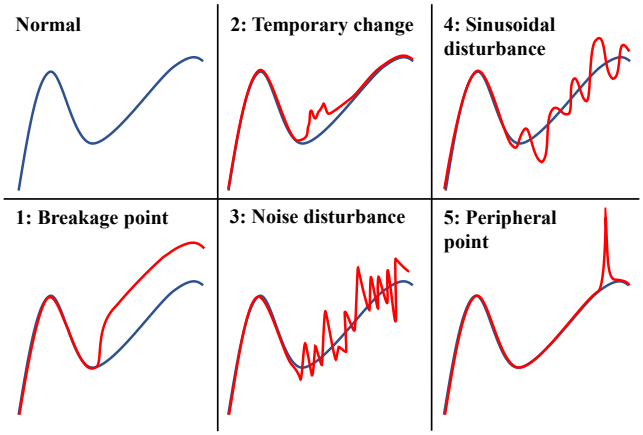

Figure 4: Description of fault types

Atomic anomalies are cases with deviant values for one variable while aggregate anomalies result from groups of variables deviating as a collective. They are contextual outliers (faults 1, 2 & 5), and collective outliers (faults 3 & 4) present on some process steps and not during the whole processing. Fault 1 is a breakage point, creating deviant cycle. Fault 2 is a temporary change in value with a return to a regular level after several time steps. Faults 3 and 4 are similar to additive noise and sinusoidal disturbances respectively, which are innovational outliers creating a trend change. Fault 5 is a peripheral point, which is an independent data point literally outlying, resulting from a sudden rise in value (peak). Fig. 5 illustrates the 5 fault types.

For all fault types, faults are introduced on one process step and each fault type occur on more at least 2 different variables but not simultaneously. Thus, for one faulty data sample, only one of the 11 variables can be faulty, with the same fault type as well as with different fault types. For each fault type, the step on which the fault occurs is chosen randomly. Thus, faults do not systematically occur on the same step.

The methods are implemented in Python 3.8 with the deep learning library Keras of TensorFlow 2.4.0, scikit-learn 0.24.1 library and Ubuntu 18.04 operating system. Experiments are conducted on a personal computer with Linux 16 CPU cores @ 2.4 GHz , 125 Go RAM and a GPU card with 3840 cores @ 1.58 GHz , 12 Go RAM.

B. Neural networks configuration

The following six deep learning models are tested in the experiment: FDC-CNN, Attention FDC-CNN, LSTM-AD, EncDec-AD, CSAE-AD, and CDSAE-AD. For each of these neural networks, the parameters where selected after testing various configurations (number of layers, number of filters/cells/nodes per layer) and the configuration with the best result chosen.

For FDC-CNN (resp. Attention FDC-CNN), two convolutional layers with 128, 256 filters (resp. 32, 64 filters) coupled with two max pooling layers (resp. attention layer after the last convolutional layer). The receptive field is (3×11) for the convolutional layers, the pool size is (2×1) and the stride 1. The attention is Luong-style attention. The fully connected layer has 100 nodes. The sigmoid activation is

used for the last layer and ReLU activation for the remaining layers.

For LSTM-AD and EncDec-AD, two layers of 512 LSTM cells. LSTM-AD has sigmoid activation between layers. For both models, the cells have tanh activation.

For CSAE-AD and CDSAE-AD, two convolutional layers with 128 and 64 filters, followed by two deconvolutional layers with 64 and 128 filters and the last layer is a convolutional layer with 1 feature map. The receptive field is (3×11) for the first convolutional layer and the stride 1 for all layers. The sigmoid activation is used for the last layer and ReLU activation for the remaining layers.

C. Other configurations

- Dataset partitioning: The training and testing sets are divided according to the ratio 8:2. A 5-fold cross validation is used during the training phase.
- Weight optimization: The kernel initializer is the glorot uniform. The Adaptive Moment Estimation (Adam) [24] optimizer is used for the training with a learning rate of 10-3. The batch sizes and epochs are 64 and 100, respectively. The cost functions are the Binary Cross Entropy and the Mean Squared Error (MSE) for the supervised and self-supervised methods, respectively.
- Weight regularization: To prevent the models from overfitting, dropout (0.1 for CNN-based, 0.05 for other models), early stopping mechanism and kernel constraints (with maxnorm of 2 for AE-based) are implemented.

D. Evaluation metrics

Precision, recall, and F1-score are used. Accuracy is a ratio of correctly predicted observations to the total observations. Accuracy, which is the most intuitive performance measure, is not a proper metric when dealing with unbalanced dataset (where one class has many more instances than the other) [25]. Precision also known as positive predictive value, is the ratio of true faults among total detected faults while recall, also known as true positive rate, is the ratio of true faults detected among the correct predictions. The F1-score is the harmonic mean of precision and recall. It reaches its best value at 1 (perfect precision and recall) and is worst at 0. Thus, the larger the precision, recall, and F1-score are, the more accurate the given anomaly detection method is. The principal metric for the anomaly detection performance will be F1-score. The metric values are calculated as follows:

$$Precision = TP/(TP + FP), \quad (10)$$

$$Recall = TP/(TP + FN), \quad (11)$$

$$F1 = 2*Precision*Recall/(Precision + Recall), \quad (12)$$

where *TP, FP,* and *FN* are true positives, false positives (type I error), and false negatives (type II error) respectively.

E. Results and discussion

Table 1 presents the results of comparison between the six aforementioned deep learning based anomaly detection methods. All the terms are the best results for the anomaly detection models, obtained with optimized hyperparameter values (best values for learning rate, batch size, number of epochs, dropout obtained using scikit-learn's GridSearchCV) and several trainings. The results in Table 1 are presented as percentages (except for training time) for more convenience.

- Firstly, the supervised methods with CNN based approach demonstrate over 95% F1-score and achieve the best detection performance. Of these, Attention FDC-CNN has the highest overall F1-score (95.5%) and only slightly outperforms the second best (FDC-CNN) by 0.4%. The AE based methods show good results and with an overall F1-score over 88%, they are relatively close in performance to the supervised methods. The LSTM based methods have the least results. With overall F1-score less than 85%, they underperform with regards to the other models.
- Secondly, when focusing on each fault individually, it can be noticed that faults are not detected the same. FDC-CNN has the highest F1-score for faults 1 and 4 (99.3% and 93.9%) while Attention FDC-CNN has the highest F1-score for faults 2, 3 and 5 (95.0%, 95.0% and 100%). This shows that models can be differentiated on their detection performances per fault.

TABLE 1: DETECTION PERFOMANCE

Method	Fault 1			Fault 2			Fault 3			Fault 4			Fault 5			Overall			
	P(*)	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	T
FDC-CNN	**98.9**	**99.7**	**99.3**	92.7	95.9	94.3	92.6	93.7	93.1	92.8	**95.1**	**93.9**	92.8	97.0	94.9	94.0	96.3	95.1	**32s**
Attention FDC-CNN	92.9	97.4	95.1	92.8	**97.4**	**95.0**	92.5	**97.6**	**95.0**	92.6	92.4	92.5	**100**	**100**	**100**	94.1	**97.0**	**95.5**	77s
LSTM-AD	96.9	75.5	84.9	96.5	75.5	84.7	96.3	75.5	84.6	96.3	75.5	84.6	99.2	75.5	85.7	97.1	75.5	84.9	769s
EncDec-AD	98.7	73.7	84.4	96.6	73.7	83.6	95.0	73.7	83.0	94.9	73.7	82.9	96.8	73.7	83.7	96.4	73.7	83.5	760s
CSAE-AD	97.6	82.6	89.5	**98.3**	82.6	89.7	97.2	82.6	89.3	97.1	82.6	89.3	**100**	82.6	90.5	98.1	82.6	89.7	52s
CDSAE-AD	98.3	80.2	88.3	**98.3**	80.2	88.3	**97.8**	80.2	88.1	**97.8**	80.2	88.1	99.9	80.2	89.0	**98.4**	80.2	88.4	144s

(*): P=Precision, R=Recall, F=F1-score, T=Training Time

- Thirdly, it can be noticed that the supervised methods have better recall than the self-supervised methods and thus the former better classify normal data samples (few type II error). Attention FDC-CNN has the highest overall recall (97.0%), showing its great ability to accurately classify normal data samples. On the other hand, the self-supervised methods have better precision than the supervised methods, hence demonstrating the better capacity of the former to detect correct faults (few type I error). Particularly, the AE based methods have the highest overall precision (98%), showing their great ability to correctly detect faults, with CDSAE-AD having the best performances with regard to precision.
- Lastly, FDC-CNN has the shortest training time (32s), due to the small number of convolutions determined by the large receptive fields. CSAE-AD has a relatively short training time too (52s), due to its ability to quickly find a local minimum during the learning phase thus achieving a high training speed amongst the other models. The LSTM based models have the longest training times, which are 8 to 16 times longer than the other models.

Regarding the above results analysis, we can say :

- The supervised anomaly detection methods give the best performances in overall detection and training time with Attention FDC-CNN having slightly better results than FDC-CNN. When having a fully labelled dataset and/or for processes requiring a minimal number of type II errors, these supervised learning methods are recommended and will yield the best results.
- The AE-based methods give second best performances in detection and training time and the best performances among self-supervised methods and have results not very far from those of the supervised methods with CSAE-AD having slightly better results than CDSAE-AD. When having a dataset with normal samples only and/or for processes requiring a minimal number of type I errors, these self-supervised learning methods are recommended and will give the best results.
- The LSTM-based methods give the least performances both in detection and training time with LSTM-AD having slightly better results than EncDec-AD.

V. Conclusion

Various neural networks like CNNs, LSTMs and AEs, which work effectively in fields as diverse as image recognition and natural language processing have been studied to propose methods to accurately detect anomalies on raw industrial data. In this paper, a comparative evaluation of some deep learning-based anomaly detection methods on multidimensional time series has been proposed. Having presented and discussed the results of our evaluation on deep learning anomaly detection techniques, it is important to spot out its limitations. The evaluation implemented in this comparative analysis is meant for multivariate time series data. It does not claim to do an evaluation of state-of-the-art deep learning anomaly detection techniques. The choice of the detection methods for the analysis was done on the basis of their industrial applicability on semiconductor multivariate data and our knowledge of the basic neural networks behind the methods.

The results analysis reveals that both supervised and unsupervised methods are performant and have an interest of use. Depending on the data and/or fault nature, one of the methods more than the other will yield great detection performances. With the plurality of data in the semiconductor industry, (multiple equipment types and a multitude of recipes per equipment), the best performances of each model can be exploited to develop a global adaptative model, based on the studied deep learning methods. It is important to underline how the usage of such a "clean" dataset as the simulated semiconductor dataset, for training and testing can result in a lower degree of generalizability of the conclusions. Future studies can be conducted by testing the models on real semiconductor data. The simulated data combined with real data can serve for data augmentation to train the neural networks, boosting their performances and the generalization capacities. Also, future work can be done to determine the minimum quantity of data necessary for each model to give good detection performances. The usage of these models for real time detection on multivariate time series can also be investigated.

Acknowledgment

We would like to thank one of our colleagues, Laurent Bidault, for providing the semiconductor simulation dataset and for insightful discussions on deep learning models.

References

[1] P. Patel, M.I. Ali, A. Sheth, "From Raw Data to Smart Manufacturing: AI and Semantic Web of Things for Industry 4.0," *IEEE Intell. Syst*., vol. 33, no 4, pp. 79–86, 2018. DOI: 10.1109/MIS.2018.043741325

[2] E. L. Park, J. Park, J. Yang, S. Cho, Y.-H. Lee, and H.-S. Park, "Data based segmentation and summarization for sensor data in semiconductor manufacturing," *Expert Syst. Appl*., vol. 41, no. 6, pp. 2619–2629, 2014. DOI: 10.1016/j.eswa.2013.11.001

[3] F. J. Anscombe, "Graphs in Statistical Analysis," *Am. Stat.*, vol. 27, no. 1, pp. 17-21, 1973. DOI: 10.1080/00031305.1973.10478966

[4] J. Moyne, J. Samantaray, M. Armacost, "Big Data Capabilities Applied to Semiconductor Manufacturing Advanced Process Control," *IEEE Trans. on Semi. Manuf*., vol. 29, no. 4, pp. 283 - 291, 2016. DOI: 10.1109/TSM.2016.2574130

[5] S.R. Saufi, Z.A.B. Ahmad, M.S. Leong, M.H. Lim, "Challenges and Opportunities of Deep Learning Models for Machinery Fault Detection and Diagnosis: A Review," *IEEE Access*, vol. 7, pp. 122644–122662, 2019. DOI: 10.1109/ACCESS.2019.2938227

[6] A. Thieullen, M. Ouladsine, J. Pinaton, "Application of Principal Components Analysis to Improve Fault Detection and Diagnosis on Semiconductor Manufacturing Equipment," *European Control Conference*, 2013. DOI: 10.23919/ECC.2013.6669553

[7] A. Thieullen, M. Ouladsine, J. Pinaton, "Application of PCA for efficient multivariate FDC of semiconductor manufacturing

equipment," *Advanced Semi. Manuf. Conf.*, 2013. DOI: 10.1109/ASMC.2013.6552755

[8] J. Marino, F. Rossi, M. Ouladsine, J. Pinaton, "Gaussian Time Error: A new index for fault detection in semiconductor processes," *American Control Conference*, 2016. DOI: 10.1109/ACC.2016.7525416

[9] S.-K. S. Fan, S.-C. Lin, and P.-F. Tsai, "Wafer fault detection and key step identification for semiconductor manufacturing using principal component analysis, adaboost and decision tree," *J. Ind. Product. Eng.*, vol. 33, no. 3, pp. 151–168, 2016. DOI: 10.1080/21681015.2015.1126654.

[10] S. Mahadevan, S. L. Shah, "Fault detection and diagnosis in process data using one-class support vector machines," *J. Process Control.*, vol. 19, no. 10, pp. 1627–1639, 2009. DOI: 10.1016/j.jprocont.2009.07.011

[11] Q. P. He and J. Wang, "Principal Component based K-Nearest-Neighbor Rule for Semiconductor Process Fault Detection," *Proc. Amer. Control Conf.*, pp. 1606–1611, 2008.

[12] M. Goldstein, S. Uchida, "A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data," *PLoS ONE*, 2016. DOI: 10.1371/journal.pone.0152173

[13] R. Chalapathy, S. Chawla, "Deep Learning for Anomaly Detection: A Survey," 2019. arXiv: 1901.03407

[14] Y. LeCun et al., "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989. DOI: 10.1162/neco.1989.1.4.541.

[15] K. B. Lee, S. Cheon, C. O. Kim, "A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes," *IEEE Trans. Semi. Manuf.*, vol. 30, no. 2, pp. 135–142, 2017. DOI: 10.1109/TSM.2017.2676245.

[16] E. Kim , S. Cho , B. Lee, M. Cho, "Fault Detection and Diagnosis Using Self-Attentive Convolutional Neural Networks for Variable-Length Sensor Data in Semiconductor Manufacturing," *IEEE Trans. Semi. Manuf.*, vol. 32, no. 3, 2019. DOI: 10.1109/TSM.2019.2917521

[17] Z. Lin et al., "A structured self-attentive sentence embedding," *Proc. Int. Conf. Learn. Represen*t., 2017.

[18] S. Hochreiter, J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[19] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, "Long Short Term Memory Networks for Anomaly Detection in Time Series," *European Symposium on Artificial Neural Networks*, 2015.

[20] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, G. Shroff, "LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection," *ICML*, 2016.

[21] G. E. Hinton, R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[22] K. Chen, J. Hu, J. He, "Detection and Classification of Transmission Line Faults Based on Unsupervised Feature Learning and Convolutional Sparse Autoencoder," *IEEE Trans. On Smart Grid*, vol. 9, no. 3, pp. 1748–1758, 2016. DOI: 10.1109/TSG.2016.2598881

[23] H. Lee, Y. Kim, C. O. Kim, "A deep learning model for robust wafer fault monitoring with sensor measurement noise," *IEEE Trans. Semi. Manuf.*, vol. 30, no. 1, pp. 23–31, 2017. DOI: 10.1109/TSM.2016.2628865.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[25] L.A. Jeni, J.F. Cohn, F. De la Torre, "Facing imbalanced data recommendations for the use of performance metrics," *Int. Conf. on Affective Computing and Intelligent Interaction*, 2013. DOI: 10.1109/ACII.2013.47

[26] M. Sugawara, "Plasma etching : fundamentals and applications," *Oxford Science Publications*, 1998.

[27] J. Xu et al., "Anomaly Detection on Electroencephalography with Self-supervised Learning," *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 363-368, 2020. DOI: 10.1109/BIBM49941.2020.9313163

[28] A. Ng, "Sparse autoencoder," *CS294A Lecture Notes, Stanford Univ.*, pp. 1–19, 2011.

[29] J. Ngiam et al., "On optimization methods for deep learning," *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, pp. 265–272, 2011.