computers

MDPI

Article

Isolation Forests and Deep Autoencoders for Industrial Screw Tightening Anomaly Detection

Diogo Ribeiro [1], Luís Miguel Matos [1], Guilherme Moreira [2], André Pilastri [3] and Paulo Cortez [1,*]

[1] ALGORITMI R&D Centre, Department of Information Systems, University of Minho, 4804-533 Guimarães, Portugal; id6336@alunos.uminho.pt (D.R.); luis.matos@dsi.uminho.pt (L.M.M.)
[2] Bosch Car Multimedia, 4705-820 Braga, Portugal; guilherme.moreira2@pt.bosch.com
[3] EPMQ-IT CCG ZGDV Institute, 4804-533 Guimarães, Portugal; andre.pilastri@ccg.pt
* Correspondence: pcortez@dsi.uminho.pt

Abstract: Within the context of Industry 4.0, quality assessment procedures using data-driven techniques are becoming more critical due to the generation of massive amounts of production data. In this paper, we address the detection of abnormal screw tightening processes, which is a key industrial task. Since labeling is costly, requiring a manual effort, we focus on unsupervised detection approaches. In particular, we assume a computationally light low-dimensional problem formulation based on angle–torque pairs. Our work is focused on two unsupervised machine learning (ML) algorithms: isolation forest (IForest) and a deep learning autoencoder (AE). Several computational experiments were held by assuming distinct datasets and a realistic rolling window evaluation procedure. First, we compared the two ML algorithms with two other methods, a local outlier factor method and a supervised Random Forest, on older data related with two production days collected in November 2020. Since competitive results were obtained, during a second stage, we further compared the AE and IForest methods by adopting a more recent and larger dataset (from February to March 2021, totaling 26.9 million observations and related to three distinct assembled products). Both anomaly detection methods obtained an excellent quality class discrimination (higher than 90%) under a realistic rolling window with several training and testing updates. Turning to the computational effort, the AE is much lighter than the IForest for training (around 2.7 times faster) and inference (requiring 3.0 times less computation). This AE property is valuable within this industrial domain since it tends to generate big data. Finally, using the anomaly detection estimates, we developed an interactive visualization tool that provides explainable artificial intelligence (XAI) knowledge for the human operators, helping them to better identify the angle–torque regions associated with screw tightening failures.

Keywords: autoencoder; deep learning; Industry 4.0; isolation forest; one-class classification; unsupervised learning

check for updates

Citation: Ribeiro, D.; Matos, L.M.; Moreira G.; Pilastri A.; Cortez, P. Isolation Forests and Deep Autoencoders for Industrial Screw Tightening Anomaly Detection. *Computers* **2022**, *11*, 54. https://doi.org/10.3390/computers11040054

Academic Editor: Osvaldo Gervasi

Received: 24 February 2022
Accepted: 6 April 2022
Published: 8 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The Industry 4.0 revolution forced most companies to adapt to a more fierce and competitive market where having optimized productive processes can dictate whether a company is successful or not. Within this context and in particular for the assembly industrial sector, there is a pressure to reduce of costs whilst increasing efficiency. Indeed, with the introduction of smart sensors and actuators, several modern factories use autonomous or semi-autonomous robots capable of assisting human operators during production processes. This occurs because some assembly tasks are hard to fully automate and thus it is important to have a human-in-the-loop. An example of such task is the bonding process between plastic parts or between plastic parts and electronic components. These are usually achieved via welding, riveting, gluing, or by simply screwing parts together (which is the focus of this paper).

Mechanically combining parts using screws is usually achieved with the help of human-operated hand-held screwdrivers. Over time, these machines collect thousands of real-time data points which are later used for quality assessment. Although the collected data might vary from factory to factory, or from handheld screwdriver manufacturers, most assume the minimum collection of angle–torque pairs. On process completion, these pairs are used to plot a tightening curve, which is presented to the operator on a monitor above the station. The operator is then prompted to infer on the process correctness using the available data and a quality estimation produced by the handheld screwdriver machine software. Despite defects being rare in this domain, some units require a more thorough inspection to prevent faulty units from moving to the next assembly stations. Examples of faulty processes are crossed or damaged threads, cracked or split parts and others. To better understand the main reasons why a screwing cycle is deemed faulty, factories make use of the collected data to build a defect catalog.

Due to the dynamic nature of the screw tightening process, which includes the introduction of different models to be assembled over time, there is a constant need to update defect catalogs. Although defect catalogs are empirically proven to work, they use a rather static and meticulous data curation process that involves manual labor and screwdriver software updates. Such rigid defect catalogs are unsuitable for industries that produce millions of fastening movements each day. Thus, there is an opportunity to develop a highly automated screw tightening inspection system by adopting machine learning (ML) algorithms. While there is vast body of knowledge covering anomaly detection in several domains, the advancements in the application of ML to fastening processes is rather scarce. To the best of our knowledge, there are only a few studies which can relate to our approach (as shown in Section 2).

In this work, we assume an anomaly detection task that consists in the automatic detection of screw tightening process failures by employing unsupervised ML algorithms. The goal is to feed these algorithms with only normal examples, creating data-driven models that tend to trigger high anomaly scores when faced with "abnormal" data (i.e., outside the learned normal input space). This task is non-trivial due to several reasons. First, as already mentioned, the screw tightening process is dynamic (e.g., assembly of new products); thus, the ML models need to be continuously updated. Second, the data is highly imbalanced, where only a tiny fraction of the examples correspond to failures (e.g., 0.2%). Third, data is generated with a high velocity, thus resulting in big data that requires a high computational effort by some ML algorithms.

In our previous work [1], we performed an initial exploration of a low-dimensional input approach (angle–torque pairs) for anomaly detection for a small subset of the available industrial data (corresponding to two days of production). Promising results were obtained by two unsupervised learning ML algorithms: isolation forest (IForest) and deep dense autoencoders (AE). In this extended paper, we further evaluate the robustness of the proposed screw tightening anomaly detection methods by considering a wider range of assembled products (three distinct products) and a larger number of industrial records, collected during a longer time period (two months). By exploring more recent data, new and different fastening curves were introduced in our system. Using these more richer screw tightening datasets, we conduct several computational experiments, measuring the anomaly detection performance and the required computational effort of the proposed IForest and AE models. Moreover, we adopt a realistic and robust rolling window, with several training and testing updates that are simulated over time. Finally, we also present a novel dynamic anomaly detection threshold visualization tool that provides explainable artificial intelligence (XAI) knowledge the human operators, supporting the identification of specific angle–torque regions related with faulty screw processes. This paper is organized as follows. Section 2 presents the related work. Next, Section 3 describes the collected industrial assembly data, anomaly detection learning models, and evaluation procedure. Then, Section 4 presents the experimental results. Finally, Section 5 discusses the main conclusions and future work directions.

2. Related Work

Screwdriving is among the common assembly methods used in industry that have been particularly difficult to fully automate despite several continual efforts [2]. A simpler characterization of the screwing curve using the angle–torque pairs was early proposed by [3] and in our previous work [1]. Moreover, the angle–torque curve was also adopted by the ISO rotary tool evaluation standards [4]. With the lower availability of anomalous data, anomaly detection is often regarded as a one-class problem, which is a form of unsupervised learning where fault detection systems are trained only on normal data. Classic algorithms such as the local outlier factor (LOF), one-class support vector machine (OC-SVM), and the more recent tree ensemble IForest method seem to perform relatively well on many of the uses cases, as reported by [5,6]. The main drawbacks of these algorithms is that they tend to require more computational effort, also offering no online update capability, thus being less recommended to handle bigger volumes of data. Deep learning autoencoders (AE) are less affected by such computational limitations. For instance, a linear increase in the training set size typically results in an exponential computational cost growth in terms of the OC-SVM training, while for the AE model the cost increase is also linear. Moreover, the training of AEs can be sped up by adopting graphics processing units (GPU). Thus, AEs are being increasingly adopted for anomaly detection tasks [7]. To the best of our knowledge, only a small portion of research studies have employed AEs in this domain.

Considering the industrial screw tightening domain, the work of [8] demonstrated the use of Bayesian rules and distance rejection heuristics to incrementally discover new manufacturing defects, which were then clustered for interpretation purposes by a manufacturing expert. Using a similar clustering principle, [9] proposed a kernel density-based pattern architecture that can successfully identify patterns in data and correctly categorize them as good or bad. However, the proposed kernel density-based architecture is too computationally expensive when handling larger datasets, which occurs in our analyzed screw tightening industrial domain. Two supervised learning methods, namely, a linear support vector machine (SVM) classifier and an artificial neural network (ANNs), were compared on [10] on their capability to discriminate between four task states: “Successful”, “Empty”, “Failed”, and “Jammed”. Although their discrimination abilities was roughly the same, the linear SVM was slightly faster. While interesting results were achieved, this work diverts from our approach under two important aspects. First, it assumes a supervised learning approach, which requires labeled data that is often costly to collect (e.g., requiring human effort and time). Second, the work of [10] is more focused on root-cause analysis rather than providing a feasible solution to the problem. Another SVM experiment was conducted by [11], which tried different kernel functions (e.g., linear and polynomial) to monitor screwdriving processes on the cover of hard disks. Unlike our use case, this experiment relies on the driver motor data instead of the data being produced by the handheld machine. Yet, the proposed SVM approach is infeasible for our domain due to the required high computational cost. For instance, when adopting a similar SVM algorithm using a preliminary (and smaller) screw tightening dataset, the learning process halted after 40 hours of execution time. More recently, [12] designed a supervised learning experiment using pairwise data (angle–torque) and a Long Short-Term Memory (LSTM) deep learning architecture for the purpose of discriminating whether a fastening process is successful or not. While good results were obtained, and similarly to [10], the experiments relied on labeled data (with both normal and abnormal cases) that can be particularly difficult to be obtained (e.g., requiring a manual data curation effort).

3. Materials and Methods

3.1. Industrial Data

This study was conducted within a large electronics manufacturer that supplies assembled components (mainly automotive instrument clusters) to some of the most recognized brands in the automobile industry. The assembly of a cluster is an extensive process with multiple assembly stages. In this work, we focus on one of the final phases in the process,

where the plastic housing is combined with either the printed circuit boards (PCB) or other plastic parts. Bonding plastics or electronics to plastics can be achieved via a multitude of techniques that involve glue, welding, or the use of screw fasteners. For the remainder of the article, we optimize the validation process of screw tightening processes, which makes use of threaded fasteners as the bonding mechanism for the units.

Validating a fastening procedure can be challenging, since it involves several real-time variables. The manufacturing experts take their knowledge into account and leverage the information provided by the handheld driver manufacturers to overcome most of the problems inherent to the usage of this bonding technique. Despite the efforts to mitigate most failure modes with approaches such as sequence based fastening, some deviations still occur on the shop floor and the control mechanisms must be able to signal them fast and accurately.

The plant standardizes most production lines to follow the most efficient layout in terms of assembled clusters throughput per hour. Individually, each assembly station follows a strict set of rules that check for violations in the correctness of the fastening sequence. The assembly process starts with the operator inserting the raw plastic housing in a assembly jig. This jig is designed with the "poka yoke" principle [13], which prevents wrong or misaligned parts from being loaded into the assembly jig. A scanner installed in a favorable position reads the imprinted identification code and verifies if the product successfully passed the previous checkpoints and if the station is capable of executing this assembly stage. The correct screw tightening program is then loaded onto the handheld tool. This stage is fundamental because different variants of the same part can be produced on the same machine where the threaded holes can be of different dimensions or be located in different places. The settings specified in this program are also used as a baseline to compare against the real values produced by the machine. The operator then guides the handheld driver to a feeder which is always on and not controlled by the program. Once a screw is loaded on the screwdriver bit, the operator is instructed to follow a predefined sequence with the aid of instructions carefully illustrated on a monitor located above the assembly station. Each inserted screw results in a good or fail (GoF) status message, presented on the screen which indicates whether the fastening succeeded or not. Simultaneously, the produced data is stored locally on a `.csv` file. Depending on the result, two different courses of action can be performed: on failure—the operator is instructed to stop the procedure; on success—the data is uploaded to a remote server where it will be thoroughly analyzed by an expert tool that compares the produced data against a defect catalog. If no defect is detected, the operator is instructed to proceed to the next unit. This new control flow differs from the one presented in our past work [1], as the manufacturing experts deemed it more important to detect false negatives (assembly cycles considered good but are not) than further analysis of faulty units that would have to restart the assembly process.

Regardless of the outcome, new files are made available on a remote data-server (separate from the one running the expert tool) and are then subjected to Extract, Transform, and Load (ETL) processes to make data easier to be worked on. Up until this stage, we have no control over the generated data stream. We are currently reading this pre-processed data using `.parquet` files partitioned by date but this process is undergoing some structural changes which will fundamentally change the way we access new data, making it available through Solace, an event broker tool [14].

Table 1 outlines some of the variables collected for each assembly process. The granularity of the data collated depends on a diverse set of factors, such as the type of the machine in use or the type of product being produced. Some machines do not even support the computation of the variables collected once per fastening (e.g., DTM based attributes). Nevertheless, all machines support the collection of angle and torque values. For each i process, there is a real-time generation of several $k \in \{1, 2, 3, \ldots, K_i\}$ values, where K_i denotes the total number of observations where each part number can produce a different K_i value. For each unique combination of i and k values, the machine collects hundreds of angle ($\alpha_{i,k}$) and torque ($\tau_{i,k}$) measurements. It should be highlighted that while there is

no direct temporal variable in the collected attributes, the angle ($\alpha_{i,k}$) attribute can be used as a sequential temporal measure of the fastening cycle. In effect, the angle represents the rotation made by the screw during the fastening process. Thus, as the angle value increases, the closer the screwing cycle is to reaching its end, which is often marked by an abrupt increase/decrease of the torque value (e.g., screw head fastened to the surface).

Table 1. Description of the screw industrial data variables.

Variable	Description
	Hundreds of values (k) per screw fastening (i):
profile_angle ($\alpha_{i,k}$)	Value for the angle (e.g., −208.1 degrees)
profile_torque ($\tau_{i,k}$)	Value for the torque (e.g., 35.6 Ncm)
profile_gradient	Torque gradient for two consecutive angle values (e.g., 20)
	Four values per fastening (one for each step):
profile_stepnr	Screwing step number ($\in\{1, 2, 3, 4\}$)
screw_total_angle	Total step angle (e.g., 990 degrees)
screw_total_torque	Total step torque (e.g., 39.3)
	One value per fastening (i):
screw_dtm_clamp_angle	Value for actual DTM Clamp angle (e.g., 79.1)
screw_dtm_clamp_torque	Value for actual DTM Clamp torque (e.g., 30.2)
screw_timestamp	Timestamp (e.g., "2020-10-08 06:30:51")
part_number	Product family identifier (e.g., "2222111")
serial_number	Product serial number (e.g., "11111")
screw_number	Screw number ($\in\{1, 2, \ldots, 8\}$)
screw_energy	Total energy required (e.g., 19,959)
screw_total_angle	Total angle (e.g., 2993 degrees)
screw_total_torque	Total torque (e.g., 30,000 Ncm)
screw_gof (y_i)	Screwing process "Good" (1) or "Fail" (0)

A typical screwdriving cycle is characterized by four stages as per described in Table 2. Should a process be successful (Figure 1), the unit needs to undergo all steps and meet the transition conditions of each sequentially. During the initial stage (step 0), the screwdriving machine rotates counterclockwise in an attempt to latch to the screw head. Although this step is already part of the assembly process it is not relevant for the current analysis and some machines do not even report it. The remaining three steps represent mechanical milestones for the correct fixation of a screw to a plastic housing. There are some specific situations where a successful procedure skips or adds an additional step. The main reason for this behavior is related to the mechanical properties of the bonding units and the capabilities of each assembly station. For example, the DTM attribute is not calculated if the handheld driver does not support such a feature. For all others, the tool estimates the clamping angle and torque for the fastening and then compares them to the actual values (represented by *screw_dtm_clamp_angle and* screw_dtm_clamp_torque). These are some of the computations the assembly machine executes to assess whether a process was finalized successfully or not, returning a GoF label (attribute *screw_gof*).

Despite using one-class algorithms and selecting the angle–torque pairs as our input features, the *screw_gof* variable, denoted as y_i for the i-th screw tightening process, is required during the data preparation step. Not only is it used to separate good from bad fastening cycles, it also serves as our target variable during the model evaluation stage. These labels are provided by the screwdriver and are computed by comparing the curve signature against a predefined set of static rules. Although these rules are created and tested by manufacturing experts, some cycles end up being misclassified. These rare occurrences are usually reported and handled as fast as possible. However, and considering the nature of this big data problem, it is not time- and computationally feasible to verify all assembly cycles that are part of our dataset. Instead, we assume that these occurrences

are very rare and thus that they do not impact on our anomaly detection training and evaluation procedures.

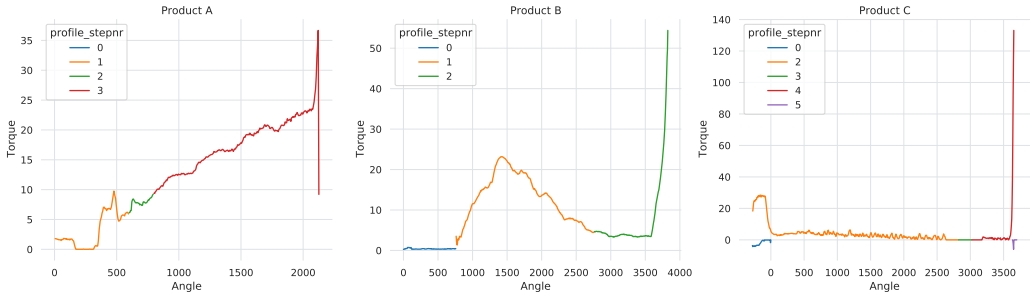

Figure 1. Example of normal ("Good") screw tightening cycles for three different products.

Table 2. The fastening process steps.

Step	Description	Transition Condition
Simple Torque Step (1)	Serves as a transition to the next step when the thread starts to be formed	Achieve either the transition torque or angle within the parameterized time range.
Angle Step (2)	Fixed number of turns conducted	Min Angle < Angle Target < Max Angle within stipulated time.
Torque Step (3)	Apply a fixed torque value	Min Torque < Torque Target < Max Torque within stipulated time.
Seating Control Step (4)	Apply a torque value from the screw seating value to the part	Clamping Torque < Max Seating Torque; Min Total Angle < Total Angle < Max Total Angle; Min Total Torque < Total Torque < Max Total Torque.

Table 3 describes the more recent and larger dataset that is explored in this paper. In the table, the product names were anonymized for confidentiality reasons (termed here as A, B, and C). These were selected based on their availability and in an attempt to cover a diversity within the universe of products fabricated by the analyzed manufacturer. Each product is composed of multiple *part_numbers* and is comprised of multiple fastening cycles that share similar characteristics (e.g., angle–torque signature). For each individual family, in this paper we collected two months worth of data (from February to March 2021), which result in a total of 23,790 unique serial numbers and roughly 26.9 million individual observations (67,337 fastening cycles times 400 data points per screw). As the research literature suggests, datasets in this domain are usually highly unbalanced. In our case, there is only a tiny percentage of faulty processes (e.g., 0.2% for Product A). Moreover, on average, there are around $\overline{K_i}$ = 400 records (angle–torque pairs) per cycle.

Table 3. Statistics of the most recently collected datasets.

	Unique Part Numbers	Unique Serial Numbers	Num. Good Screws	Num. Bad Screws	Ratio
Product A	52	6189	8512	19	0.002
Product B	5	13,160	45,662	99	0.002
Product C	3	4441	13,163	102	0.008
Total	60	23,790	67,337	220	

3.2. Anomaly Detection Angle–Torque Approach

Given the important of the screw tightening anomaly detection task, several experiments were previously conducted by the manufacturing experts. A large range of experiments were held, including the application of batch process monitoring procedures [15].

However, this industrial anomaly detection task is non-trivial. Unlike other types of processes (e.g., chemical reactions), there can be "normal" abrupt changes in the torque values, as shown in Figure 1 in terms of the final angle–torque curve for Product A. In effect, inefficient results were achieved when adopting the batch process monitoring attempts. Nevertheless, in one of their attempts, the experts conducted a principal component analysis (PCA) that supported our assumption that angle–torque pairs are fundamental to evaluate screwing processes. Moreover, when we started our research, we performed an initial set of experiments using a larger number of input variables (from Table 1). Yet, the ML models (e.g., LOF, IForest, AE) obtained much worst class discrimination results while requiring an expensive computational effort. In some cases, such as when using LOF, the computational training process even halted due to an out-of-memory issue. Based on these results, we then opted to focus on the low-dimensional angle–torque input data approach, which provided better results with a much lower computational effort. Thus, in this work we developed two ML algorithms (IForest and AE) that make use of two data features (angle and torque values) as the input values, producing then an anomaly decision score (d). It should be noted that during training, the ML algorithms are only fed with normal instances.

Considering that each fastening cycle is comprised of $k \in \{1, 2, \ldots K_i\}$ angle–torque observations, the output of each detection model will contain $d_{i,k}$ anomaly decision scores, where i denotes the i-th screw being evaluated. The overall decision score (d_i) is computed by averaging each individual score such that $d_i = \sum_i d_{i,k}/K_i$. For each unique i value, the resulting score (d_i) can be compared to a target label of a test set (y_i), making the computation of the Receiver Operating Characteristic (ROC) curve [16] possible. Given that human operators require a class label, a fixed Th threshold value is adopted such that for the i-th output is considered anomalous when $d_i > Th$. The overall procedure used to compute the final screw classification score (d_i) is shown in Figure 2.

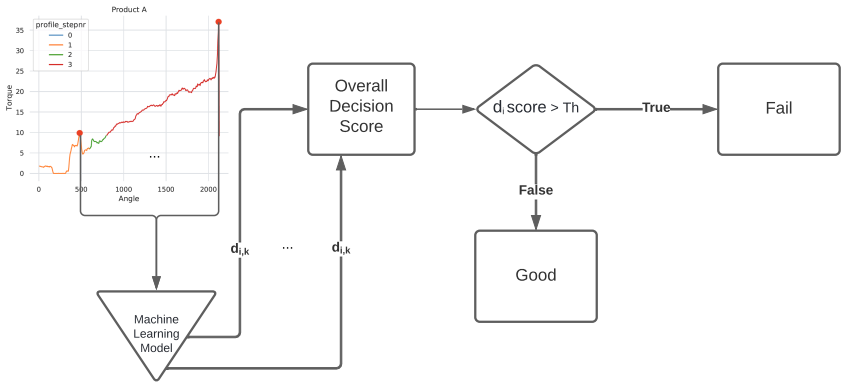

Figure 2. Schematic of the overall screw tightening anomaly detection process.

Selecting the most appropriate Th requires either domain knowledge or a semi-automatic selection of the best specificity–sensitivity trade-off of a ROC curve generated using a validation set (with labeled data). In this work, we assume the usage of domain knowledge, as provided by the screw tightening human operators. To support the expert Th value selection, we propose a specialized XAI interactive visual tool that includes a threshold selection mechanism. The tool allows a finer control over the threshold selection process, allowing experts to manually specify individual thresholds for different families of products and to easily identify problematic angle–torque regions within the screw tightening profile curves.

The data collected during each individual assembly process contains 44 unique attributes spanning three granularity levels. Table 1 summarizes the 16 attributes related to the actual fastening process, while the remaining 28 features are used by the assembly management systems. As described in Section 3.1, each process is composed of multiple observations of angle–torque pairs ($\alpha_{i,k}$-$\tau_{i,k}$) and the torque gradient between two consecutive

angle values is grouped by step number. Each step is comprised of multiple fine-grained observations, the total angle and torque for the given step. Subsequently, each step is grouped under a specific fastening identifier, which, alongside the aforementioned data, reflects the whole industrial screw tightening process.

While there is a large number of attributes, several experiments were conducted beforehand by manufacturing experts to try and address the problem currently being studied. Experts designed a multitude of experiments using multivariate techniques which proved to be inefficient and incapable of discriminating good from bad screwing cycles. For confidentiality reasons the results of such experiments can not be made public. In one of their runs, the conducted Principal Component Analysis (PCA) further supported our hypothesis that angle–torque pairs contain the bulk of the information necessary to evaluate screwing processes. Additionally, during the set-up of our previous work [1] we composed a group of experiments with multiple combinations of variables and encoding techniques (such as one-hot encoding) which not only increased the overall model complexity but proved to add no extra capacity to our models. As such, and given the previously obtained results using older screw tightening data it was found that the low-dimensional angle–torque input data provided a better anomaly detection performance while requiring much less computational effort. Thus, all screw anomaly methods described in this work assume just two input values: the simpler angle–torque pairs $(\alpha_{i,k}, \tau_{i,k})$ for each (i, k) example.

3.3. Machine Learning Models

In our previous work [1], we compared several ML algorithms for screw tightening anomaly detection: three unsupervised algorithms, namely, LOF, IForest, and a deep AE; and the Random Forest (RF) supervised learning algorithm.

The *LOF* is a density-based algorithm [5] that heavily depends on K-nearest neighbors [17]. It is designed as local because it depends on how well isolated an object is from the surrounding neighborhood. Instead of classifying an object as being an outlier or an inlier, an outlier factor is assigned describing up to which degree the object differs from the rest of the dataset. It should be noted that LOF can be used to detect both local and global outliers. Moreover, it can be trained with multi-class instances. In this paper, in order to provide a fair comparison, we only use normal examples to fit the model, where the outlier factor score is directly used as the anomaly decision score ($d_{i,k}$). For benchmarking purposes, we also considered the RF model, which requires labeled training data and that is build in terms of a large number of decision trees, which are then combined together as an ensemble to get more accurate and reliable predictions [18].

In this work, we focus more on two unsupervised learning models that provided competitive results in our previous work [1]: IForest and AE. All anomaly detection methods were coded using the Python language. The LOF, IForest and RF methods assumed the `scikit-learn` module implementation [19], while the AE was developed using the `TensorFlow` library [20]. In order to provide a fair comparison, whenever possible we adopted the default hyperparameter values assumed by the `scikit-learn` and `TensorFlow` implementations. Before feeding the models, the angle–torque training data were normalized by using the `MinMaxScaler` procedure of the `sklearn` library, scaling all inputs within the range [0, 1].

Isolation forest is an anomaly detection algorithm which, as the name indicates, uses the isolation principle to classify data as anomalous rather than modeling normality. It leverages the power of smaller decision trees combining them into a single, more capable architecture. As it makes no use of labeled data, it is an unsupervised model. It operates based on the principle that anomalies are few and numerically different from normal instances. This forces anomalous instances to be isolated easier (separated from all other instances) than normal data. Using these principles as foundation, a tree structure is created in an attempt to isolate every single instance by applying multiple splits with random parameters and then assessing on their normality. Anomalous observations with fewer attributes capable of describing them tend to be positioned closer to the root of the tree.

This structure is usually regarded as an *Isolation Tree* (or *iTree*) and is the main component of anomaly detection of this algorithm. As such, an IForest [21] is an ensemble of iTrees that when combined can discriminate between good and bad data. Figure 3 summarizes the working principle of this algorithm, where the data points with shorter average path lengths are indicated as anomalies. On the other side of the spectrum, data points with bigger average path lengths are considered normal. Data points which fall in between these too extremities are classified as potential anomalies. These distances are computed and expressed by `scikit-learn` as an anomaly score ranging from $\hat{y}_{i,k} = -1$ (highest abnormal score) to $\hat{y}_{i,k} = 1$ (highest normal score), where $\hat{y}_{i,k}$ denotes the IForest output for the k-th angle–torque pair of the i-th screw tightening instance. In order to compute an anomaly probability, we rescale the IForest score by computing $d_{i,k} = (1 - \hat{y}_{i,k})/2$.

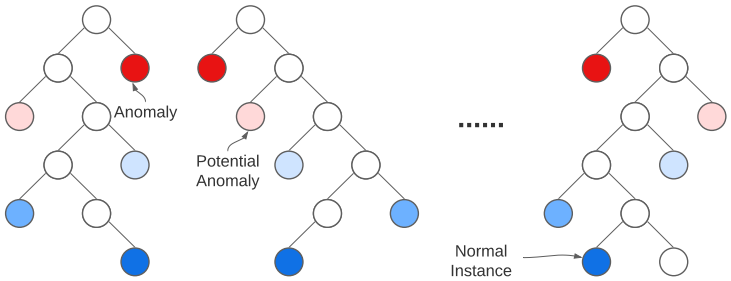

Figure 3. The Isolation Forest working principle, adapted from [22].

Unlike the IForest, an AE is a type of artificial neural network that learns normality rather than computing abnormality scores. The AE is comprised of two main stages: an initial stage where it learns a representation for a set of data (encoding), typically by reducing the input data (the number of features describing the input data) and is usually unaffected by noise; and a decoding stage, in which the model tries to reconstruct the input signal. This neural network architecture imposes a bottleneck out of the encoding stage (which forces dimensionality reduction), resulting in a compact knowledge image of the original input, usually referred to as latent space. When applied to the specific case of anomaly detection, the architecture accepts normal data as input and then attempts to produce $(\hat{\alpha}_{i,k}, \hat{\tau}_{i,k})$ outputs which should be identical to the input pair $(\alpha_{i,k}, \tau_{i,k})$, where $\alpha_{i,k}$ denotes the angle and $\tau_{i,k}$ the torque for the k-th measurement of the i-th screw tightening instance. To ascertain the quality of this reconstruction, on each new (i, k) instance we compute its Mean Absolute Error (MAE) [6], formulated as follows:

$$MAE_{i,k} = (|\alpha_{i,k} - \hat{\alpha}_{i,k}| + |\tau_{i,k} - \hat{\tau}_{i,k}|)/2 \quad (1)$$

The reconstruction MAE is used as the anomaly decision score $d_{i,k} = MAE_{i,k}$ for each input instance, where higher reconstruction errors stand for a higher abnormality probability. As previously mentioned, the architecture we developed assumes $(\alpha_{i,k}, \tau_{i,k})$ pairs as input and produces two output nodes which form a fully connected structure including a stack of layers for both stages. All intermediate hidden layers are activated using the ReLU function, contrasting with the output nodes which assumes a logistic function (all $\alpha_{i,k}$ and $\tau_{i,k}$ values are normalized $\in[0, 1]$ as previously described). To address and reduce the internal covariate shift, which occurs when the input distribution of the training and test set differ but the output label distribution remains intact, we apply a Batch Normalization (BN) layer, discarding the need for dropout layers. In fact, batch normalization also normalizes the layer inputs for each batch of data that passes through it [23]. The development of this network architecture, all its assumptions and decisions resulted from several trials, conducting during preliminary experiments using older screw tightening data. Each experiment was conducted by imposing one bottleneck layer with 1 hidden node and a varying range of additional hidden layers. The best performing

architecture was achieved when the AE was composed of 20 layers (input, *BN*, and output layers) as shown in Figure 4. Additionally, all models were trained using the Adam optimization algorithm (which proved to be slightly better than SGD in our test case), using the previously described loss function (MAE), a total of 100 epochs and early stopping (with 5% of the training data being used as the validation set).

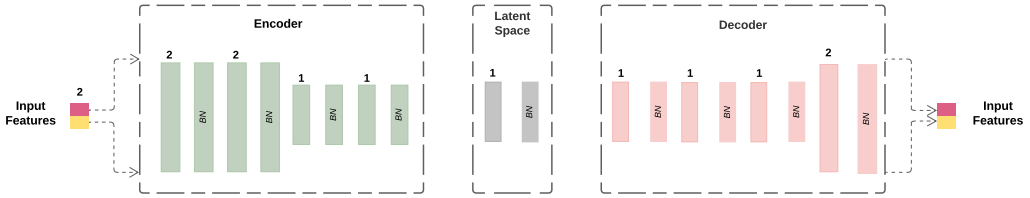

Figure 4. The adopted deep dense AutoEncoder (AE) structure (the numbers denote the number of hidden units per layer and the term BN represents a Batch Normalization layer).

As explained in Section 2, the AE model can be adapted to dynamic environments when there are frequent data updates. In [24], two neural network learning modes were compared: reset and reuse. When new data arrives and the neural network is retrained, the former assumes a random weight initialization, while the latter uses the previously trained weights as the initial set of weights.

3.4. Evaluation

With the goal of assessing a robust performance, for both IForest and AE methods we assumed the realistic rolling window procedure [24,25]. This procedure includes a fixed size window for the training (W) and test (T) data and that rolls over time by using a jumping step (of size S), thus generating several training and testing model iterations (Figure 5). At the end, there are a total of $U = (D - (W + T))/S$ iterations, where D denotes is the total number of screw cycles available in the analyzed data. During each rolling window iteration execution, only the training data is used to fit the ML model, thus the test set is considered as "unseen" data, meaning that it is only used for anomaly detection performance evaluation purposes. The goal of this evaluation is to realistically measure the behavior of the anomaly detection methods through time, assuming a continuous model retraining usage. Using the most recently collected data (regarding the three analyzed products), and after consulting the manufacturing experts, we assume a total of $U = 8$ standard rolling window train and test iterations by fixing the following values: Product A–W = 6809, T = 851, and S = 106; Product B–W = 36,529, T = 4566, and S = 570; Product C–W = 10,530, T = 1316, and S = 164 (all values related to numbers of screw fastening processes). We particularly note that these values are much higher than the ones employed in our previous work [1]. This occurs because previously we have collected only two days of screw industrial data, while our product datasets (presented in this work) have a substantially higher number of data records (67,337 fastening cycles related with two months).

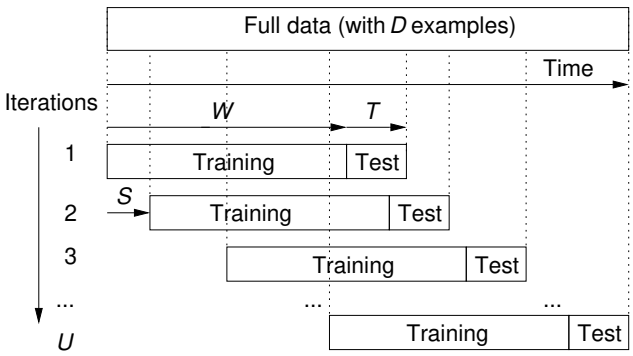

Figure 5. Schematic of the adopted rolling window procedure.

To evaluate the anomaly detection performance, we adopt the Receiver operating characteristic (ROC) curve [16], which produces a visual representation of the ability of a binary classifier to distinguish between normal and anomalous data as its discrimination threshold varies. In the case of both the standard and static training rolling window procedures, the ROC curves are computed by using the target labels and the anomaly scores ($d_{i,k}$) for the T tightening test examples of each rolling window iteration. The overall discrimination ability is then measured by the Area Under the Curve ($AUC = \int_0^1 ROC\, dTh$) and the Equal Error Rate (EER). As argued in [26], the AUC measure has several advantages. For instance, the measurement of quality is is unaffected by balancing issues in the dataset (as previously explained, our datasets are highly unbalanced). Additionally, interpreting its results is fairly easy and can be categorized as follows: 50%—performance of a random classifier; 60%—reasonable; 70%—good; 80%—very good; 90%—excellent; and 100%—perfect. The EER is used to predetermine the threshold value for which the false acceptance rate and false rejection rate is equal [27]. Under this criterion, the lower the value, the more accurate the classification. Given that multiple AUC and EER values are generated for each rolling window iteration, the experimentation results are aggregated by computing their median values and using the Wilcoxon non-parametric statistic test [28] to check whether the paired median differences are statistically significant.

All experiments were executed on a M1 Pro processor with integrated GPU. For each ML model, we collected the computational effort, measured in terms of the rolling window average preprocessing, training, and testing (inference) time (all in seconds *s*), per anomaly detection method. This is of particular interest to the manufacturing experts that need to deploy the developed systems in an industrial context with near real-time requirements.

4. Results

4.1. Previous Experiments

In our previous work [1], three unsupervised (one-class) learning algorithms were selected for an empirical comparison: LOF, IForest, and an AE. These methods were only trained on normal instances. For benchmarking purposes, we also selected the popular Random Forest (RF) method. However, we note that RF is a supervised learning method that, contrary to the one-class methods, requires labeled data for the training procedure. Thus, RF was trained with both normal and abnormal angle–torque instances. The dataset used for training corresponds just to two days of screw tightening processes, collected in November of 2020, and it included a total of 2,853,967 entries related with $N = 6162$ fastening cycles. The adopted evaluation scheme was a rolling window, under the parameterization: $W = 5000$, $T = 500$, and $S = 33$ (thus producing $U = 20$ model updates).

The obtained results are summarized in Table 4 and show that both the RF and IForest achieved the highest discrimination scores with a median AUC score of 99%. The LOF has shown the worst discriminating performance (AUC of 80%). When comparing both AE learning modes, reuse and reset, the former provides slightly better median AUC results and lower average computational training times (explained by the usage of the early stopping procedure).

Table 4. Rolling window screw tightening anomaly detection results (best values in **boldface** font; selected models in *italic* font).

AUC		Train Time (s)	Inference Time (s)
LOF	0.80	**18.30**	0.12
RF	**0.99** ⋆	25.33	0.12
IForest	**0.99** ⋆	168.18	0.08
AE reset	0.93 ⋆	29.64	**0.04**
AE reuse	0.95 ⋆	23.00	**0.04**

⋆—Statistically significant under a paired comparison with LOF.

It should be noted that the RF requires labeled data and it was included in the previous experiments for benchmarking purposes. The IForest and AE results (particularly the reuse mode) are very similar to the RF performance and these methods present the advantage of not requiring labeled data.

4.2. Experiments with Larger and More Recent Data

In this section, we further compare the performance of the two unsupervised anomaly detection methods that previously provided the best results (Section 4.1): IForest and AE reuse. For such purpose, we adopt more recently collected data and that corresponds to a larger time period (two months). Moreover, the analyzed datasets are associated with three distinct assembled product families, which tend to produce different screw tightening profile curves (as shown in Figure 1).

Table 5 summarizes the rolling window evaluation results for the three products and the two selected learning architectures (IForest and AE reuse), while the individual AUC and EER values (obtained for each rolling window iteration) are shown in Figure 6. The results attest an excellent discrimination level that was achieved by the two learning methods for all three products, almost reaching the perfect AUC value (in effect, we had to increase the AUC precision values to three digits in order to distinguish them from the maximum AUC = 1.0 value; Table 5). In terms of the discrimination power, both methods present very similar performances. For instance, the maximum median AUC difference is just 0.7 percentage points). Furthermore, as shown in Figure 6, the IForest and AE reuse curves are very close (e.g., the maximum individual difference for the same iteration is just 3.8 percentage points (second iteration for Product A). Turning to the EER criterion, it clearly favors the AE models, which produce lower values for all three products when compared with the IForest model. Moreover, the AE is computationally much faster when compared with IForest. In effect, the latter method requires on average around 2.7 times more computational effort for training and around 3.0 times more inference time (to detect if a screwing procedure is anomalous). Given that the screw tightening domain often generates big data, this computationally light property of the AE reuse method is highly valuable.

Table 5. Rolling window screw tightening anomaly detection results (best values in **boldface** font).

		AUC	EER	Train Time (s)	Inference Time (s)
IForest	Product A	**0.998**	0.840	115.933	0.018
	Product B	**0.999**	0.899	1664.067	0.029
	Product C	**0.999**	0.917	321.862	0.019
AE	Product A	0.996	**0.793**	**61.250**	**0.005**
	Product B	0.996	**0.832**	**842.869**	**0.012**
	Product C	**0.999**	**0.593**	**76.467**	**0.006**

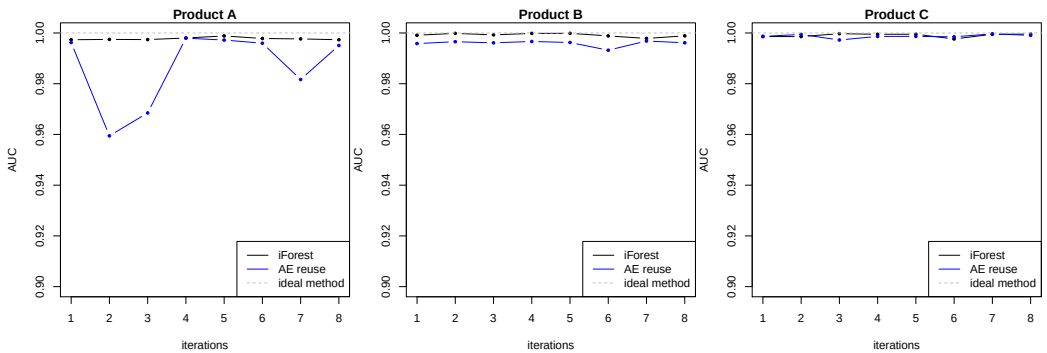

Figure 6. AUC for each rolling window iteration.

While the ERR criterion can be used to set classification score thresholds, the assumed balance between false acceptance and rejection rate is not applicable to the analyzed industrial screw tightening context, where a higher false acceptance rate is less costly than failing to accept a valid unit. To better explain the anomaly detection results to the human operators, we have developed an interactive tool that assumes a dynamic threshold selection (Figures 7 and 8). The tool works by first loading a trained anomaly detection model. Then, each time a new fastening cycle is analyzed (the inference process), the operator can manually perform a real-time experiment with different threshold values (Th) by using a slider button (shown at the bottom of the figures). Each time a new threshold is selected, angle–torque regions with high anomaly scores (i.e., $d_{i,k} > Th$) are highlighted (yellow colored points in Figures 7 and 8). As shown in the graphs, a lower threshold produces a higher angle–torque region selection area.

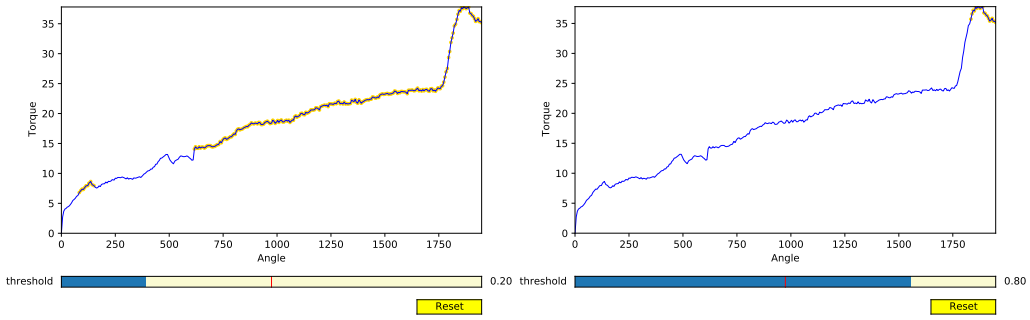

Figure 7. Example of the dynamic threshold tool usage for a Product A torque reached too soon error.

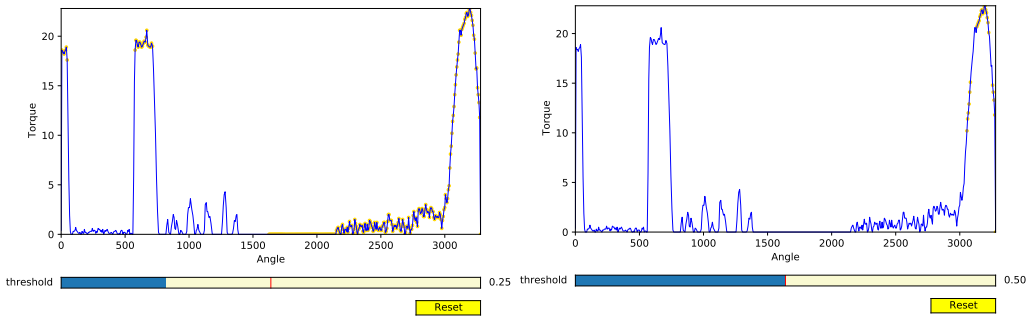

Figure 8. Example of the dynamic threshold tool usage for a Product B stripped screw error.

The purpose of the interactive tool is twofold. First, it can be used by the human operators to fix the final threshold, which is adopted to signal all abnormal screw cases. Second, it can be used to better reason about the ML anomaly detection decisions. In effect, the interactive threshold graphs works as an explainable artificial intelligence (XAI) tool, helping in the identification of the angle–torque regions that were responsible for the anomaly, which can potentially support the identification of screw anomaly causes.

We experimented the tool with both IForest and AE resuse methods. In general, the same threshold level produced the same anomaly angle–torque region selection. To simplify the visualization, the two selected XAI screw anomaly examples are related with the AE reuse model. Figure 7 demonstrates a torque reached too soon error. Although the tightening curve resembles the one presented in Figure 1, the model correctly detected that the torque values for the corresponding angle were reached too soon (which usually indicates a stripped plastic thread fault). In Figure 8, a different type of failure mode was detected by the AE model. The unexpectedly steep torque values at the beginning of the process point at a mating problem between the screw and plastic threads. This seems to be confirmed by the expected values of torque being reached at a higher angle (meaning the screwdriver had to perform a few more rotations). The apparent loss of torque in the

middle of the curve signature is normal for some parts, given that the tolerances for the screw cavities are rather high on this product.

5. Conclusions

The Industry 4.0 revolution is currently impacting several industry sectors. In particular, manufacturing industries are rapidly adopting such technologies, using sensor data to build intelligent systems capable of providing detailed insights on a multitude of manufacturing processes. In this paper, we addressed a major assembly manufacturer for the automotive sector that adopts semi-automated industrial processes, involving both human operators and robotic machines. After the assembly process was complete, a two-step validation approach was conducted by both parties, with the exception of faulty units. In the event of a failure (signaled by the machine), human assessment is not required being the unit sent to the rework station or the lab, if it is a recurrent failure. Good processes are compared to a defect catalog that is composed of failures not detected by the machine or the operator. This catalog follows a strict set of rules imposed by the manufacturing experts, thus tending to be rather rigid, due to the human effort required for any updates. Thus, there is room for improvement by using machine learning algorithms, creating a data-driven model that can more quickly adapt to changes in the assembly environment (e.g., assembly of new products).

In this paper, we explored two unsupervised approach methods for anomaly detection in fastening cycles. We first compared the proposed methods (IForest and AE) with two baseline methods (an local outlier factor and a supervised Random Forest), using two days of industrial data collected in November 2020. Since competitive results were obtained, we further evaluated the IForest and AE methods on a more recent and larger data, regarding three types of assembled products and two months of production, from 12 February to March 2021. In total, 67,337 fastening cycles and roughly 26.9 million angle–torque pair observations were considered in this second evaluation stage. Overall, an excellent anomaly discrimination detection performance was obtained by both methods, with the AE requiring much less computational effort. Finally, we designed an interactive visualization tool that provides XAI knowledge to the human operators, helping them to better identify the angle–torque areas associated with anomalies.

In future work, we intend to deploy the proposed methods in a real production environment, monitoring their performance during a prolonged period of time and also obtaining a valuable feedback from the screw tightening human operators. Finally, we plan to study the effect of other deep AE architectures, for instance by incorporating Long Short Term Memory (LSTM) layers that can capture sequential data patterns [12].

Author Contributions: Conceptualization, D.R., L.M.M., G.M., A.P. and P.C.; methodology, L.M.M., A.P. and P.C.; software, D.R. and L.M.M.; validation, G.M., A.P. and P.C.; formal analysis, P.C.; investigation, D.R. and L.M.M.; resources, D.R., L.M.M. and G.M.; data curation, D.R.; writing—original draft preparation, D.R. and L.M.M.; writing—review and editing, L.M.M., A.P. and P.C.; visualization, D.R., L.M.M., P.C.; supervision, G.M. and P.C.; project administration, G.M. and A.P.; funding acquisition, G.M. and P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by: European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project n 39479; Funding Reference: POCI-01-0247-FEDER-39479]. The work of Diogo Ribeiro is supported the grant FCT PD/BDE/135105/2017.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Acknowledgments: The authors wish to thank the anonymous reviewers for their helpful comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ribeiro, D.; Matos, L.M.; Cortez, P.; Moreira, G.; Pilastri, A.L. A Comparison of Anomaly Detection Methods for Industrial Screw Tightening. In Proceedings of the Computational Science and Its Applications-ICCSA 2021-21st International Conference, Cagliari, Italy, 13–16 September 2021; Proceedings, Part II; Gervasi, O., Murgante, B., Misra, S., Garau, C., Blecic, I., Taniar, D., Apduhan, B.O., Rocha, A.M.A.C., Tarantino, E., Torre, C.M., Eds.; Springer: Berlin, Germany, 2021; Volume 12950, pp. 485–500. [CrossRef]
2. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv.* **2009**, *41*, 15:1–15:58. [CrossRef]
3. Bickford, J. *Handbook of Bolts and Bolted Joints*; Taylor & Francis: Abingdon, UK, 1998.
4. *ISO 5393:2017*; Rotary Tools for Threaded Fasteners—Performance Test Method. International Organization for Standardization: Geneva, Switzerland, 2017.
5. Breunig, M.M.; Kriegel, H.; Ng, R.T.; Sander, J. LOF: Identifying Density-Based Local Outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; Chen, W., Naughton, J.F., Bernstein, P.A., Eds.; ACM: New York, NY, USA, 2000; pp. 93–104. [CrossRef]
6. Alla, S.; Adari, S.K. *Beginning Anomaly Detection Using Python-Based Deep Learning*; Apress: Berkeley, CA, USA, 2019. [CrossRef]
7. Zhou, C.; Paffenroth, R.C. Anomaly detection with robust deep autoencoders. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 665–674.
8. Ferhat, M.; Ritou, M.; Leray, P.; Le Du, N. Incremental discovery of new defects: application to screwing process monitoring. *CIRP Ann.* **2021**, *70*, 369–372. [CrossRef]
9. Diez-Olivan, A.; Penalva, M.; Veiga, F.; Deitert, L.; Sanz, R.; Sierra, B. Kernel Density-Based Pattern Classification in Blind Fasteners Installation. In H*ybrid Artificial Intelligent Systems*; Martínez de Pisón, F.J., Urraca, R., Quintián, H., Corchado, E., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 195–206.
10. Matsuno, T.; Huang, J.; Fukuda, T. Fault detection algorithm for external thread fastening by robotic manipulator using linear support vector machine classifier. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 3443–3450. [CrossRef]
11. Ponpitakchai, S. Monitoring Screw Fastening Process: an Application of SVM Classification. *Naresuan Univ. Eng. J. NUEJ* **2016**, *11*, 1–5. [CrossRef]
12. Cao, X.; Liu, J.; Meng, F.; Yan, B.; Zheng, H.; Su, H. Anomaly Detection for Screw Tightening Timing Data with LSTM Recurrent Neural Network. In Proceedings of the 2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), Shenzhen, China, 11–13 December 2019; pp. 348–352. [CrossRef]
13. Shimbun, N.K. *Poka-Yoke: Improving Product Quality by Preventing Defects*; CRC Press: Boca Raton, FL, USA, 1989.
14. Solace. Advanced Event Broker. An Event Mesh for Connected Enterprises. December 2021. Available online: https://solace.com/ (accessed on 23 February 2022).
15. MacGregor, J.F.; Nomikos, P. Monitoring batch processes. In *Batch Processing Systems Engineering*; Springer: Berlin, Germany, 1996; pp. 242–258.
16. Fawcett, T. An introduction to ROC analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [CrossRef]
17. Fix, E.; Hodges, J.L. Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *INternational Stat. Rev. Rev. Int. Stat.* **1989**, *57*, 238–247. [CrossRef]
18. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
19. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
20. Gulli, A.; Pal, S. *Deep Learning with Keras*; Packt Publishing Ltd.: Birmingham, UK, 2017.
21. Liu, F.T.; Ting, K.M.; Zhou, Z. Isolation Forest. In Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), Pisa, Italy, 15–19 December 2008; IEEE Computer Society: Washington, DC, USA, 2008; pp. 413–422. [CrossRef]
22. Regaya, Y.; Fadli, F.; Amira, A. Point-Denoise: Unsupervised outlier detection for 3D point clouds enhancement. *Multim. Tools Appl.* **2021**, *80*, 28161–28177. [CrossRef]
23. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML 2015), Lille, France, 6–11 July 2015; Volume 37, pp. 448–456.
24. Matos, L.M.; Cortez, P.; Mendes, R.; Moreau, A. Using Deep Learning for Mobile Marketing User Conversion Prediction. In Proceedings of the International Joint Conference on Neural Networks, IJCNN 2019, Budapest, Hungary, 14–19 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8. [CrossRef]
25. Tashman, L.J. Out-of-sample tests of forecasting accuracy: an analysis and review. *Int. J. Forecast.* **2000**, *16*, 437–450. [CrossRef]
26. Pereira, P.J.; Cortez, P.; Mendes, R. Multi-objective Grammatical Evolution of Decision Trees for Mobile Marketing user conversion prediction. *Expert Syst. Appl.* **2021**, *168*, 114287. [CrossRef]
27. Reich, C.L.; Vijaykumar, S. A Possibility in Algorithmic Fairness: Can Calibration and Equal Error Rates Be Reconciled? In Proceedings of the 2nd Symposium on Foundations of Responsible Computing, FORC 2021, Virtual, 9–11 June 2021; Ligett, K., Gupta, S., Eds.; Schloss Dagstuhl-Leibniz-Zentrum für Informatik: Wadern, Germany, 2021; Volume 192, pp. 4:1–4:21. [CrossRef]
28. Hollander, M.; Wolfe, D.A.; Chicken, E. *Nonparametric Statistical Methods*; John Wiley & Sons: Hoboken, NJ, USA, 2013.