An ensemble unsupervised method for anomaly detection in industrial production data

DACIAN GOINA
WEST UNIVERSITY OF TIMIȘOARA
Faculty of Mathematics and Computer Science
Big Data – Data Science, Analysis and Technologies
Email: dacian.goina00@e-uvt.ro

Abstract
Working machines are largely used in industrial environments for production of items and generate lots of data following these processes. The proper operation of machines influence the production output, thus detection of anomalies in machines activities is a crucial thing for avoiding awful outcomes. This paper present an ensemble unsupervised anomaly detection method able to handle aspects such as efficiency and data volume. Proposed method consists of 2 stages: in the first stage, statistical-based methods are used to assign labels to input data, then second stage use feature bagging technique to create and train estimators later used for prediction.

Keywords: *unsupervised learning, anomaly detection, ensemble methods, time series processing*

Introduction

In our times, monitoring sensors which collects data are used more and more in miscellaneous fields such as agriculture, medicine, smart grids [1] etc. The proper behavior in systems activity is a crucial thing for business costs and include different implications. Detection of anomalous behavior must be considered for preventing awful outcomes. Several faults in systems activities could lead to defects and unpleasant events. For example, Aljameel et al. [2], mentioned that several faults in systems which monitor activity of oil and gas pipelines could lead to human injuries but also to affect the environment: oil leakage produce ecological disasters.

Sensors which records specific data such as energy consumption are largely used in industrial environments for production processes. Systems such as hydroforming presess and milling machines receive patterns for specific items and start execute operations to produce the specific items by themself, human interventions didn't being mandatory. During the production processes, these sensors record huge quantities of data that describe machines activity and behavior. The proper execution of working machines is a crucial thing because it affects production output quantity and quality, finally affecting the business costs.

The collected data can be used to detect anomalies in the machines activity. Usually, the recorded data is structured as time series, i.e numerical values recorded at regular equally distanced time intervals. In many cases, working machines execute a relatively small number of operations repetitively, thus there could exists a cyclic / periodic behavior in sequence of the recorded values. Due to the fact that usually the sensors collects lots of data, the employed anomaly detection method must be able to handle several aspects such algorithm efficiency and data volume. At the same time, the used method must to be effective and to provide correctness - even with high data dimension, the method must to be able to provide quality results, to overcome different issues such as curse of dimensionality [3]. This paper present an ensemble unsupervised anomaly detection method able to deal with the previous mentioned issues. The proposed method relies on various concepts such as statistical-based methods, sliding window processing, feature bagging technique and ensemble learning strategy.

The main author contributions for this work are:

1. Introduction and presentation of a new anomaly detection method approach
2. The realization and conduction of the experiments which present the usage of introduced method for real data
3. The analysis of the experiments results and a comparison between them and the results obtained with other anomaly detection specific methods

The rest of this paper is organized as follows: section Related work present state of the art works approaches for the addressed problem, Solution methodology section present and explain the theoretical aspects of the proposed solution, Experiment section presents the contucted experiments, results analysis and comparisons, finally Conclusions section present the future directions together with final conclusions of the work presented in this paper.

Related work

Aljameel et al. [2] used classic classifiers such as Random Forest, Support Vector Machines etc to detect anomalies on gas pipeline system monitoring data. The authors conducted two experiments: the first one use default parameters values for classifiers, and a second one which use optimal parameters values. Metrics such as accuracy, precision, F1 score were used to measure models performances. The presented results showed that the models from the second experiment provided better results: Support Vector Machines method provided highest scores, reaching values of 0.97 for all previous mentioned metrics. Russo et al. [4] used active learning approach together with supervised machine learning methods for anomaly detection in the data obtained from an eutrophication experiment. Using a manual labeled dataset, a small number of instances are used to train an initial model, then the model is tested on unseen data and performance metrics are computed – if the current model do not fulfill the desired performances, the labeled data is queried again and the process is repeated until the model achieve the desired performances. The authors tested the proposed approach with several supervised learning classifiers. K-nearest neighbors provided the highest performances, reaching a F1 score value of 0.98.

Radaideh et al. [5] used RNN based models to detect anomalies in the eletronics signals. Fourier spectral smoother was used to remove noise from the waveform data, then a time lag („lookback”) strategy was used to split the data into multiple segments. The resulted data segments were used to train three models: Long Short Term Memory (LSTM), Gated recurrent unit (GRU) and a convolutional LSTM (ConvLSTM). The conducted experiments concluded that ConvLSTM model provided the highest performances, reaching a value of 0.997 for R^2 metric. Yunxiao et al. [6] presented a Variational Autoencoder (VAE) model for anomaly detection in time series data. The authors used self-adversarial approach to train a VAE model, the aim being to construct a model able to recognize the normal data and to fail in recognizing anomalous data. The experimental results showed that the proposed model provided high F1 scores with values over 0.95 on multiple datasets.

Onsem et al. [7] presented a hierarchical-based approach for detection of anomalous behavior in time series data using a pattern matching strategy. Starting from a time series, use minimizing sum of squared errors to determine the best straight line which fit the series sequence values and retain the line equation coefficients. Then split the sequence into 2 subsequences and determine the best fitting lines and retain coefficients, repeat the process by a logarithmic number of times with respect to initial time series sequence length. Use the obtained coefficients to construct (define) patterns for each subsequence, then classifiy the extracted series values using a tree based structure. The proposed method was employed for several experiments: obtained results provided high scores for recall metric.

Chang et al. [8] used unsupervised classification methods for anomaly detection in time series data collected from diffusion processes from semiconductor manufacturing. Binary segmentation was employed to split the original time series into different sequences with respect to the changing points. Further, Local Outlier Factor (LOF) method was used to compute an anomaly score for each subsequence, then the resulted scores were used to train an Isolation Forest model. Experiments on different datasets were conducted in order to compare the proposed method with other specific methods such as One Class Support Vector Machines (OC SVM), Dynamic Time Warping (DTW): the proposed method provided highest accuracy and F1 score for all used datasets.

Lee et al. [9] presented the usage of clustering methods for detection of anomalous behavior in time series data obtained from a milling machine used for manufacturing processes. Gramian Angular Field (GAF) method was used to convert time series data into images which are further fitted into a Style-GAN model to generate appropriate images of normal instances. The aim was to build a model able to accurately generate images of normal time series and to fail in generating images for anomalous time series. The obtained images are represented as vectors of values, then Locally Linear Embedding reduction method was applied to convert previous mentioned vectors into 4 dimensional vectors: these vectors were used as training data for several clustering methods. HDBSCAN method provided the best performances reaching precision value of 0.77 and recall value of 0.97.

Zhou et al. [10] presented an interval-based approach for anomaly detection in the time series data. Information granulation technique is employed to split time series data into several segments, then first order difference method is applied over the obtained segments. Further, similary measurement metrics are used to compute the similarities between the new segments values. The computed similarity scores are used to define a threshold value to separate anomalies from normal instances. The proposed method was tested with synthetic and real data: the results showed that the proposed method performed better than other methods such as Local Outlier Factor (LOF) and Piecewise Aggregate Approximation (PAA) in terms of accuracy rate.

Solution methodology

The proposed method consists of two stages as presented in the figure 1. In the first stage, statistical-based methods are used to determine the distribution for each feature of the data, then assign labels (1 or 0) to the data using the previous mentioned distributions. In the second stage, the features having most values with low occurrences probabilities are selected using a moving window approach, then feature bagging technique is employed to create estimators using the selected features. Each of the created estimators is trained with the instances labeled on the first stage: for each estimator, draw a sample of labeled instances in a stratified way - ensure to select both normal and anomalous instances. Use normal instances from sample to determine centroid value and compute weighted distances between centroid and each instance from the sample, then compute the percentiles for the obtained distances. For the prediction procedure, each estimator compute the distance between his own centroid and the new instance and if the obtained value is higher than a certain percentile, then the estimator classify the instance as being anomalous. The final label is decided using a voting system between all estimators.

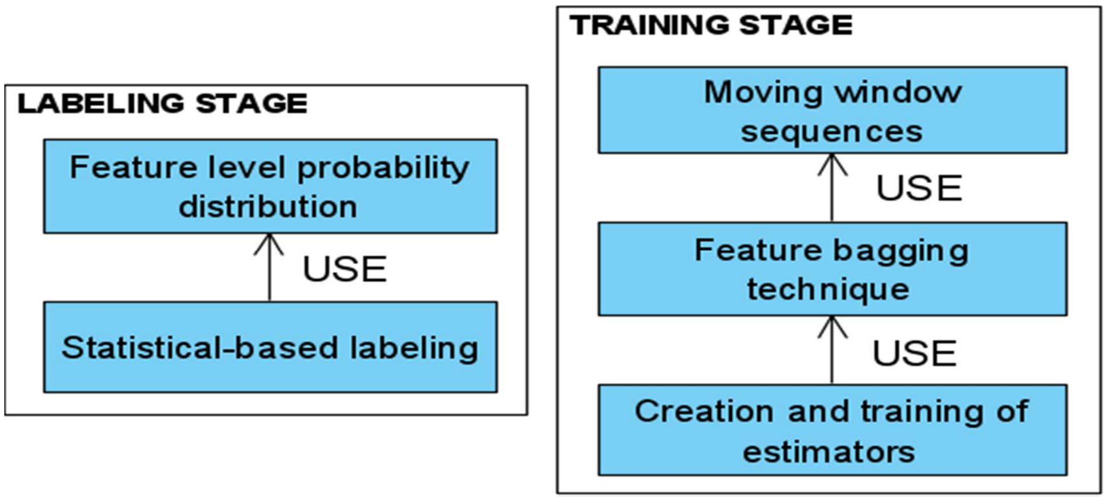
LABELING STAGE
Feature level probability distribution
USE
Statistical-based labeling
TRAINING STAGE
Moving window sequences
USE
Feature bagging technique
USE
Creation and training of estimators

Fig. 1. Proposed method – general presentation

Author: Dacian Goina

Labeling stage use statistical-based methods to assign synthetic labels to the input data. The idea starts from the vaguely definition of the anomaly: because the anomaly detection task is an unsupervised learning problem, ground-truth labels aren't provided, thus it's hard to define what exactly an anomaly is. Usually, an anomaly is defined as something that behave different from the others [11]: in statistical terms, having the probability distribution of the data, a value is considered to be anomalous / uncommon if his occurrence probability is lower than a certain threshold.

Even if the used data is directly structured as tabular data, or is represented as time series, at the end of preprocessing stage, data used is represented using several features. On this way, in order to be fitted into a classifer, the data is structured as a collection of instances, each one with certain values for a fixed set of features.

Due to the fact that is hard to work directly on high dimensional data, the distribution is determined for each feature individually. Because the estimation of probability distribution is not an easy task, a more accesible solution is the use of probability density estimation. A histogram approach is used to estimate the probabilty density: for a feature j, gather up values from all instances of the dataset and create the histogram using bins – a proper number of bins is selected using the Freedman–Diaconis rule [12]. After obtaining the bins, compute bins relative values: for each bin divide his size (number of values in the bin) to the total number of values (from all bins). Sort bins in the asceding order using the relative values and use relative values to compute a cumulative sum until a given threshold is reached: the bins whose relative values were used for the cumulative sum are considered as being low density bins. The explained process is presented in algorithm 1; consider a bin b being defined by an index b_i, lower and upper bound values b_l, b_u, number of elements in the bin b_c and bin relative value b_m.

```
**Input**: dataset $D \subseteq R^{m \times n}$, $score \in [0, 1]$
**Output**: $A = \{A_0, A_1, \dots, A_{n-1}\}$ low density bins for each feature

$A \leftarrow empty\ list$
for each feature j
    $C^j \leftarrow$ values of feature j from all instances
    $B \leftarrow get\_bins(C^j)$
    $B \leftarrow sort\_bins\_by\_relative\_values(B)$
    $sum \leftarrow 0$
    $i \leftarrow 0$
    $A_j \leftarrow empty\ list$
    $\boldsymbol{while}\ sum < score\ \boldsymbol{do}$
        $b \leftarrow B_i$    $//\ current\ bin$
        $score \leftarrow score + b_m$
        add bin $b$ to $A_j$
        $i \leftarrow i + 1$
return $A$
```

Algorithm 1: Selection of low density bins

Synthetic labeling procedure is presented in algorithm 2: an instance is considered anomalous (label value 1) if it contains at least k anomalous values; a value is considered as being anomalous if it belongs to a low density bin. The positions of anomalous values are also retained.

```
**Input**: instance $c \in D$, $A = \{A_0, A_1, \dots, A_{n-1}\}$ low density bins for each feature, $k < n$
**Output**: (state, list); state = 1 if c is labeled as anomaly, 0 otherwise

$positions \leftarrow empty\ list$
$j \leftarrow 0$
$count \leftarrow 0$
$\boldsymbol{while}\ j < n\ \boldsymbol{do}$
    $b \leftarrow bin\_of\_value(c_j)$
    $\boldsymbol{if}\ b$ in $A_j$ then
        $count \leftarrow count + 1$
        add $j$ to $positions$
    $j \leftarrow j + 1$

$\boldsymbol{if}\ count \geq k$ then
    return $(1,\ positions)$
else
    return $(0,\ empty\ list)$
```

Algorithm 2: Synthetic label assignation

Training stage use the instances with synthetic labels to create and train model estimators. Start by using the positions of anomalous values from (synthetic) anomalous instances to determine v, the vector with frequencies of positions of anomalous values; v have the same size as number of features. An exemplification plot for values from v is showed in figure 2.

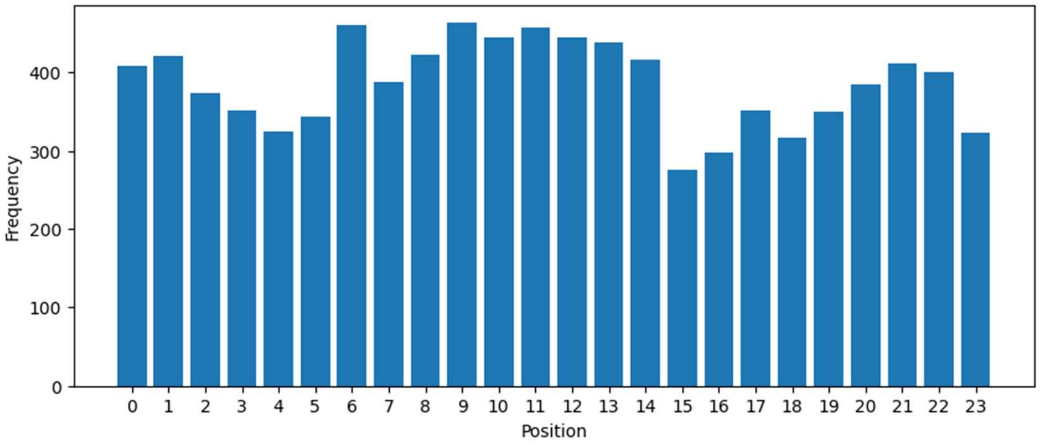

Fig. 2: Anomalous positions frequencies values (exemplification)
Author: Dacian Goina

Consider v' as relative frequencies values for values of v: $v'_j = \frac{v_j}{\sum_{i=0}^{n-1} v_i}$, $j = \overline{0, n-1}$

Further, moving window strategy is applied to generate sequences of consecutive positions from v: take windows of different lengths and generate all sequences of consecutive positions; e.g for window length $w_L = 4$ and $n = 24$ features, the obtained sequences are [0, 3], [1, 4], .., [21, 24]. Multiple window lengths should be used for more diversity: consider minimal length $l = \lfloor n \cdot 0.2 \rfloor$ and maximal length $L = \lfloor n \cdot 0.6 \rfloor$. Denote $w_{(a,b)}$ as sequence of consecutive positions values from a to b. Each $w_{(a,b)}$ item have a score computed as mean of the values from v' delimited by positions a, b:

$$S(w_{(a,b)}) = \frac{\sum_{j=a}^{b} v'_j}{b-a+1}$$

Select top N windows $w_{(a,b)}$ using the computed scores. N is the number of estimators used by the model; usually, N doesn't need to be a big value, $N = 10$ is a properly option. For creation and training of estimators: for each $w_{(a,b)}$ selected, draw a sample G of instances from synthetic labeled dataset D' in a stratified way: ensure that G contains a percentage p' of anomalous instances. Consider G' as collection of instances of G such that for each instance is taken only the sequence of values from the positions delimited by a, b; compute the centroid (mean) value μ using normal instances from G' and for each instance of G' compute the weighted distance between it and the centroid, then compute the percentiles for the obtained values. Formula (1) present the computation of weighted distance for an instance x and centroid μ. Formula (2) present the computation of weights α using v.

$$d_x = \sum \alpha_i |x_i - \mu_i| \qquad (1)$$

$$\alpha_i(w_{(a,b)}) = \frac{v_{a+i}}{\sum_{j=a}^{b} v_j}, \quad i = \overline{0, b-a} \quad (2)$$

The training procedure is presented in the algorithm 3. For model prediction procedure on a new instance a, each estimator compute the distance between his own centroid and a: if the obtained value is higher than a given percentile, then the estimator consider the instance a as being anomalous. A majority vote is applied over the labels provided by all estimators to decide the final label value for the instance a. The choice for the percentile rank is related to value of p' (percentage of anomalous instances from G); for example if $p' = 0.02$ means the estimators were trained with 2% anomalous instances, thus statistically, the distances values computed for the anomalous instances should be higher than the 98th percentile: the distances computed for anomalous instances are higher than the distances computed for normal instances. The relation between p' and percentile rank q is $q \cong 1 - p'$.

```
**Input**: $w_{(a,b)}, D', f \in [0,1]$ sample size, $p', v$
**Output**: trained estimator $W$

$distances \leftarrow empty\ list$
$G \leftarrow get\_sample(D', f, p')$
$G' \leftarrow get\_sliced\_instances(G, a, b)$
$G'_0 = \{c \in G' \mid label(c) = 0\}$
$\mu \leftarrow compute\_centroid(G'_0)$
$\alpha \leftarrow compute\_alphas(v, a, b)$
**for each** $c\ in\ G'$
        $d_x \leftarrow \sum \alpha_i |x_i - \mu_i|$
        add $d_x$ to $distances$

$P \leftarrow percentiles(distances)$
$W \leftarrow InitEstimator()$
$W_\alpha, W_\mu, W_P \leftarrow \alpha, \mu, P$        // retain weights, centroid and percentiles for later usage

return $W$
```

Algorithm 3: Creation and training procedure for model's estimator

Experiment

Several experiments were conducted to test the performance of the proposed model. The used dataset contains records of energy consumption values of a hydroforming press. The hydroforming press receive an item pattern and shape the input material into the pieces according to the provided pattern: to produce items, the press perform the same specific operation repetitively. The collected data is structured as a time series; due to the cyclic nature of the press activity, a periodic behavior could be observed in the values – fig. 3. The cycle lenght is 24, thus an instance (cycle) have 24 values (features). The cycles were extracted from the times series using the Matrix profile method [13]. From all instances (cycles), 90% was used for model training and 10% was used for testing. Table 1 present a summary related to the usage of extracted instances. The whole code was written in Python. The experiments were executed on Google Colab notebook with default (free to use) configuration.

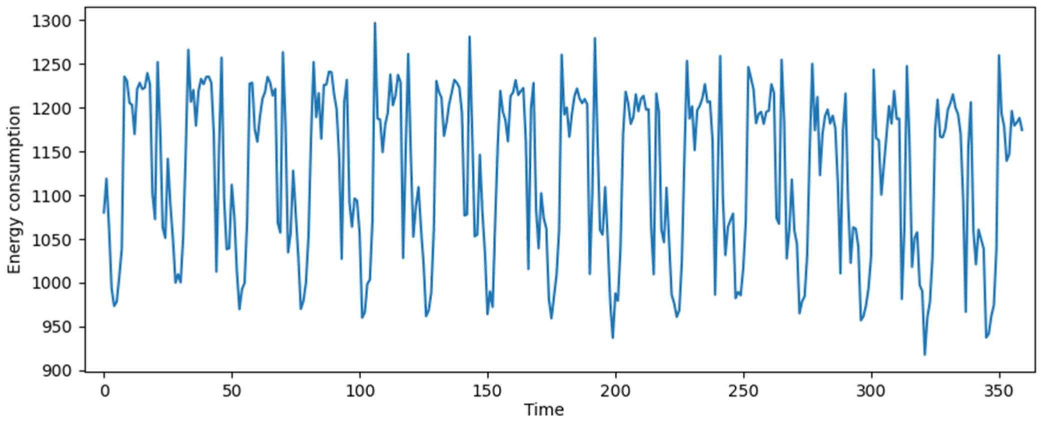

Fig. 3: Recorded energy consumption of hydroforming press

Author: Dacian Goina

No. of features	No. of instances	No of instances for model training	No of instances for model testing
24	161631	145467	16164

Table 1: Experimental data summarization

Model evaluation: ground-truth labels aren't provided, thus metrics such as accuracy, recall etc cannot be computed. To manage this issue, other anomaly detection methods are employed to perform prediction on the testing data. The logic is the followed: if the proposed model classify an instance as being anomalous maybe this outcome is not so truthful, but if other methods conclude the same (the evaluated instance is anomalous), then is a higher confidence for the instance to be truly anomalous. For each instance from the testing set, the outcome label is predicted by the proposed model but also by Isolation Forest (IF) and Stochastic Gradient Descent One Class Support Vector Machines (SGD OC SVM) models; the obtained labels are compared to check if they are all equal (matching between the predicted values). Because the used evaluation logic is related to the sets intersection, Jaccard [14] and Dice [15] scores are used to measure the similarity of the obtained sets of labels. Formulae (3) and (4) present Jaccard and Dice measures for cases with two and three sets.

$$J(A,B)=\frac{|A\cap B|}{|A\cup B|},\quad J(A,B,C)=\frac{|A\cap B\cap C|}{|A\cup B\cup C|}\qquad(3)$$

$$D(A,B)=\frac{2\cdot|A\cap B|}{|A|+|B|},\quad D(A,B,C)=\frac{3\cdot|A\cap B\cap C|}{|A|+|B|+|C|}\qquad(4)$$

Table 2 present parameters configuration for the models used in the experiments. The proposed method is referred as *EUADM (Ensemble Unsupervised Anomaly Detection Method).* For the rest of the methods, parameters names correspond to names from sklearn documentation [16]. *Contamination* and *nu* parameters refer to the percentage of anomalies to be predicted and the values were chosen such that to be closer to the percentage of synthethic anomalous instances from EUADM.

EUADM	IF	SGD OC SVM

score = 0.03; k = 12; N = 10
l = 4; L =14; f = 0.45; p' = 0.02
	n_estimators = 100
	contamination = 0.022
		nu = 0.026
		max_iter = 25

Table 2: Parameters configuration for experimental models

Tables 3 and 4 present the number of normal and anomalous instances predicted by each model: these are just raw predicted outcomes, without matching between the results. Figures 4, 5, 6, 7 present the obtained Jaccard and Dice scores for instances classified as normal / anomalous.

Method \ Percentile rank	96	97	98	99
EUADM	15485	15644	15795	15992
IF	15814			
SGD OC SVM	15797			

Table 3: Number of normal outcomes predicted by each model

Method \ Percentile rank	96	97	98	99
EUADM	679	520	369	172
IF	350			
SGD OC SVM	367			

Table 4: Number of anomalous outcomes predicted by each model

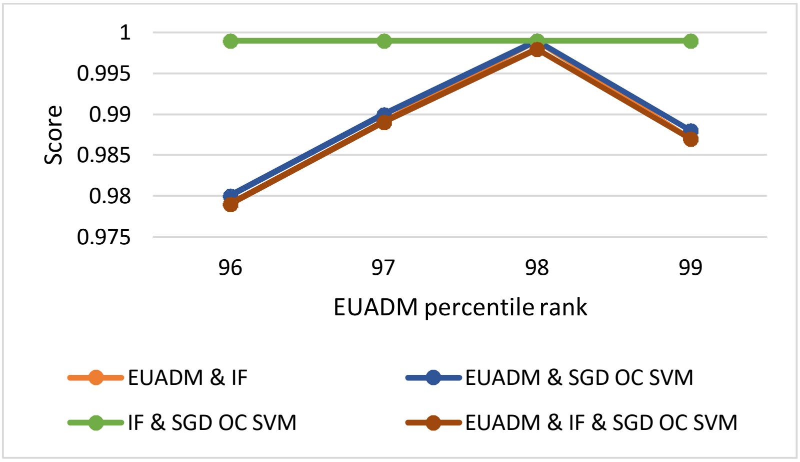

Fig. 4: Jaccard scores for instances classified as normal
Author: Dacian Goina

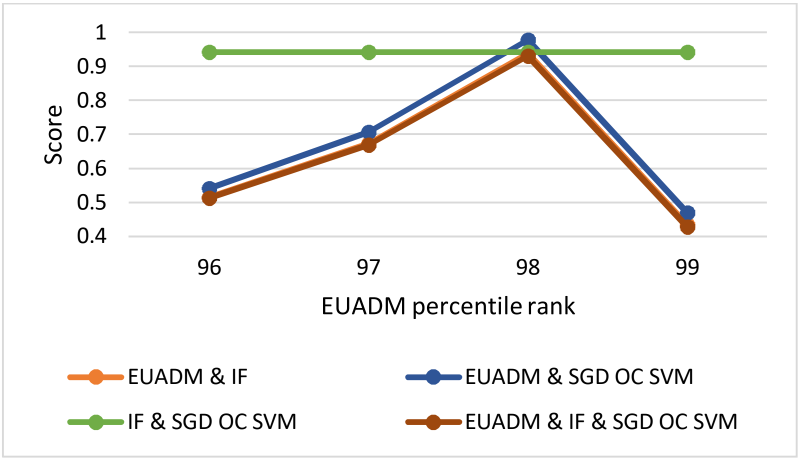

Fig. 5: Jaccard scores for instances classified as anomalous
Author: Dacian Goina

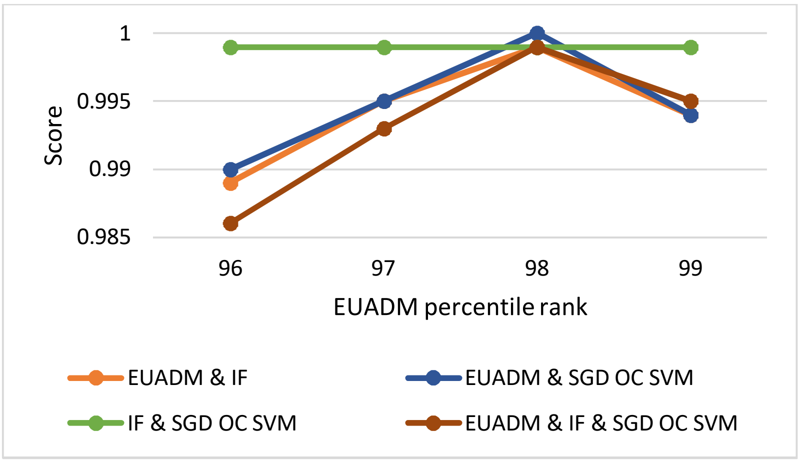

Fig. 6: Dice scores for instances classified as normal
Author: Dacian Goina

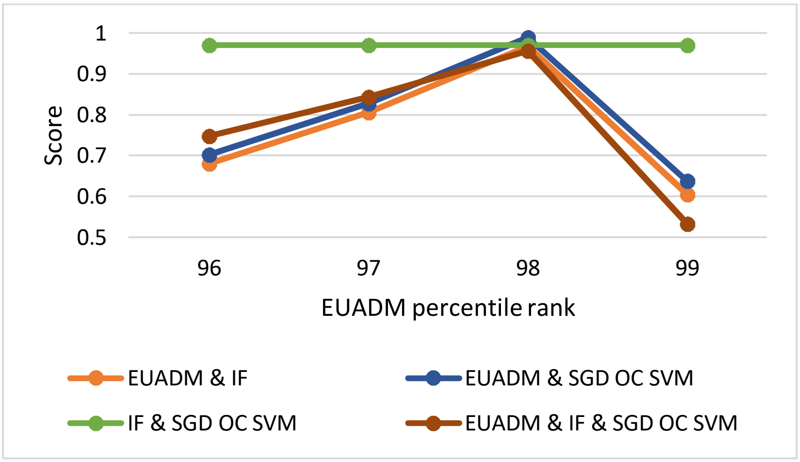

Fig. 7: Dice scores for instances classified as anomalous
Author: Dacian Goina

Results analysis: Jaccard and Dice scores obtained for instances classified as normal are higher than the scores for instances classified as anomalous, the reason is related to the fact that the number of instances classified as normal is much bigger than the number of instances classified as anomalous (tens of thousands versus hundreds - as can be observed in the tables 3 and 4), thus a small number of misclassified normal instances do not affect the score heavily. The scores obtained for instances classified as anomalous are also high, but here you can observe how crucial percentile rank is: percentile rank $q = 98$ provide the best results for all cases: this fact is related to $p' = 0.02$, the percentage of synthetic anomalous instances used for the estimators training, as explained in the theoretical section. As another observation, the proposed method EUADM have more matches in the results (higher scores) with SGD OC SVM method, in comparison with IF.

Conclusions

This paper presented an ensemble unsupervised method for detection of anomalies in production data. The proposed method consists of 2 stages: in the first stage, synthetic labels are assigned to the dataset instances using statistical-based methods. In the second stage, feature bagging technique together with synthetic labeled instances are used to create and train model's estimators. The proposed model was tested on a dataset with 161k instances and 24 features, and the obtained results were compared with the results provided by other anomaly detection methods. The obtained Jaccard and Dice scores showed that the proposed method EUADM provided high performances, but the choice of the percentile rank parameter value from the prediction method of EUADM is crucial for results quality: the advantage is that the proper percentile rank could be determined using the percentage of synthetic anomalous instances parameter from the training stage. For future work, different procedures in model creation could be tried, e.g the computation of estimators centroids using a weighted average with respect to the distribution of training samples, instead of using a simple mean as it is now. Concluding, the unsupervised anomaly detection method presented in this paper is a solution that provide high performance, closer to the results provided by other anomaly detection specific methods, and is able to handle aspects such as correctness and data volume.

Bibliography

1] P. K. Shabad, A. Alrashide and M. Osama, "Anomaly Detection in Smart Grids using Machine Learning," 2021.

2] S. S. Aljameel, D. M. Alomari, S. Alismail, F. Khawaher, A. A. Alkhudhair, F. Aljubran and R. M. Alzannan, "An Anomaly Detection Model for Oil and Gas Pipelines Using Machine Learning," *Computation,* vol. 10, no. 8, 2022.

3] E. Swartling and P. Hanna, Anomaly Detection in Time Series Data using Unsupervised Machine Learning Methods: A Clustering-Based Approach, 2020.

4] S. Russo, M. Lürig, w. hao, B. Matthews and . K. Villez, "Active learning for anomaly detection in environmental data," *Environmental Modelling and Software,* vol. 134, 2020.

5] M. I. Radaideh, C. Pappas, J. Walden, D. Lu, L. Vidyaratne, T. Britton, K. Rajput, M. Schram and S. Cousineau, "Time series anomaly detection in power electronics signals with recurrent and ConvLSTM autoencoders," *Digital Signal Processing,* vol. 130, 2022.

6] L. Yunxiao, L. Youfang, X. QinFeng, H. Ganghui and W. Jing, "Self-adversarial variational autoencoder with spectral residual for time series anomaly detection," *Neurocomputing,* vol. 458, pp. 349-363, 2021.

7] M. Van Onsem, D. De Paepe, . S. Vanden Hautte, P. Bonte, V. Ledoux, A. Lejon, F. Ongenae, D. Dreesen and S. Van Hoecke, "Hierarchical pattern matching for anomaly detection in time series," *Computer Communications,* vol. 193, pp. 75-81, 2022.

8] K. Chang, Y. Yoo and J.-G. Baek, "Anomaly Detection Using Signal Segmentation and One-Class Classification in Diffusion Process of Semiconductor Manufacturing," *Sensors,* vol. 21, no. 11, 2021.

9] T. Lee, Y. Kim, Y. Hyun, J. Mo and Y. Yoo, "Unsupervised Anomaly Detection Process Using LLE and HDBSCAN by Style-GAN as a Feature Extractor," *International Journal of Precision Engineering and Manufacturing,* vol. 25, 2023.

10] Y. Zhou, H. Ren, Z. Li and W. Pedrycz, "An anomaly detection framework for time series data: An interval-based approach," *Knowledge-Based Systems,* vol. 228, 2021.

11] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich and . K.-R. Muller, "A Unifying Review of Deep and Shallow Anomaly Detection," *Proceedings of the IEEE,* vol. 109, no. 5, p. 756–795, 2021.

12] D. Freedman and P. Diaconis, "On the histogram as a density estimator: L2 theory," *Probability Theory and Related Fields,* vol. 57, no. 4, pp. 453-476, 1981.

13] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen and E. Keogh, "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 2016, pp. 1317-1322.

14] P. Jaccard, "Comparative study of the floral distribution in a portion of the Alps and Jura," *Bulletin of the Vaudois Society of Natural Sciences,* pp. 547-579, 1901.

15] L. R. Dice, "Measures of the Amount of Ecologic Association Between Species," *Ecology,* vol. 26, no. 3, pp. 297-302, 1945.

16] "Scikit-learn," [Online]. Available: https://scikit-learn.org.