A comparative study of anomaly detection methods for gross error detection problems.

DOBOS, D., NGUYEN, T.T., DANG, T., WILSON, A., CORBETT, H., MCCALL, J. and STOCKTON, P.

2023

Supplementary materials that are held separately on the publisher's website have been appended to this file, appearing after the author responsibilities statement section of the main text.

Computers and Chemical Engineering 175 (2023) 108263

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

A comparative study of anomaly detection methods for gross error detection problems

Daniel Dobos [a], Tien Thanh Nguyen [a,*], Truong Dang [a], Allan Wilson [b], Helen Corbett [b], John McCall [a], Phil Stockton [b]

[a] National Subsea Centre, Robert Gordon University, Aberdeen, UK
[b] Accord ESL, Aberdeen, UK

ARTICLE INFO

Keywords:
Gross error detection
Machine learning
Anomaly detection
Deep Learning

ABSTRACT

The chemical industry requires highly accurate and reliable measurements to ensure smooth operation and effective monitoring of processing facilities. However, measured data inevitably contains errors from various sources. Traditionally in flow systems, data reconciliation through mass balancing is applied to reduce error by estimating balanced flows. However, this approach can only handle random errors. For non-random errors (called gross errors, GEs) which are caused by measurement bias, instrument failures, or process leaks, among others, this approach would return incorrect results. In recent years, many gross error detection (GED) methods have been proposed by the research community. It is recognised that the basic principle of GED is a special case of the detection of outliers (or anomalies) in data analytics. With the developments of Machine Learning (ML) research, patterns in the data can be discovered to provide effective detection of anomalous instances. In this paper, we present a comprehensive study of the application of ML-based Anomaly Detection methods (ADMs) in the GED context on a number of synthetic datasets and compare the results with several established GED approaches. We also perform data transformation on the measurement data and compare its associated results to the original results, as well as investigate the effects of training size on the detection performance. One class Support Vector Machine outperformed other ADMs and five selected statistical tests for GED on Accuracy, F1 Score, and Overall Power while Interquartile Range (IQR) method obtained the best selectivity outcome among the top 6 AMDs and the five statistical tests. The results indicate that ADMs can potentially be applied to GED problems.

1. Introduction

In the chemical industry, highly accurate and reliable measurements play an important role in process condition monitoring, control, and operational optimisation. Efficiency analysis and improved measurement accuracy lead not only to more profitable operations but can also be useful for detecting operational faults. Unfortunately, due to the nature of measurement, measured data inevitably contains errors from several sources such as power supply fluctuations, network transmission and signal conversion noise, analog input filtering, and changes in ambient conditions (Jordache and Narashimhan, 1999). We describe this type of error as random measurement error. It generally has a normal distribution with zero mean and known variance.

Data reconciliation aims to eliminate random errors by reconciling measurements to process constraints e.g., mass or energy balance. Data Reconciliation (DR) emerged in the mid-1960s and since then it has been applied to many areas such as the chemical and energy industries (Loyola-Fuentes and Smith, 2019). Established data reconciliation techniques use mathematical methods, such as least-squares, to adjust measurements utilising process model equations such as equilibrium equations and conservation laws (Jordache and Narashimhan, 1999). Those data points that require to be adjusted more than an expected amount are flagged up as potential errors for further investigation.

It is widely recognized that the techniques of reconciliation work under the assumption that only random errors are present in the data. If non-random errors (called gross errors, GEs) caused by, for example, instrument failure, measurement bias, or process leaks, are also present, the reconciled result can be very inaccurate and even infeasible (Jordache and Narashimhan, 1999). Detecting GEs is thus an important step before obtaining final reconciled estimates.

* Corresponding author.
E-mail address: t.nguyen11@rgu.ac.uk (T.T. Nguyen).

https://doi.org/10.1016/j.compchemeng.2023.108263
Received 8 January 2023; Received in revised form 17 March 2023; Accepted 17 April 2023
Available online 19 April 2023

In the past years, several gross error detection (GED) methods were introduced and most of them are based on statistical tests. The first statistical GED method was proposed by Reilly and Carpani (1963) in 1963. Since then, further methods such as the measurement test (Crowe and Garcia Campos, 1983), and the nodal test (Mah et al., 1976) have been proposed and widely applied. Although statistical tests are widely used by industries, two of their obvious shortcomings should be noticed. First, the statistical tests work on process data which is corrected with the help of steady-state material and balance models of the process (Jordache and Narashimhan, 1999). In the existing literature, GED and DR models work under the assumption that model equations capture the process without any mathematical error. In practice, however, the models could be inaccurate, highly likely leading to uncertainties in states which violate the underlying assumption. Moreover, statistical tests for GED only look at a snapshot in time but there are patterns in data that can help to understand how a system or a particular meter is performing over time. With the help of Machine Learning (ML) approaches, patterns in data can be discovered to obtain useful knowledge for decision-making. While improving upon statistical GEDs, new approaches from Machine Learning (ML) such as Neural Networks (Reddy and Mavrovouniotis, 1998) and ensembles of GE detectors (Nguyen et al., 2020), have also been developed and many of them have proved to be suitable for applications. From this point of view, this paper aims to explore the potential of a data-driven approach utilising ML for the problem of GED.

The basic principle of GED is a special case of the detection of outliers (or anomalies) in data analytics (Jordache and Narashimhan, 1999). Anomalies are patterns that do not conform to a well-defined notion of behaviours (Chandola et al., 2009). Anomaly detection (AD) refers to the labelling of observations as anomalous through methods, models, and items based on data (Ruff et al., 2021). Nowadays, many applications of AD can be found in diverse areas, such as cybersecurity (Patcha and Park, 2007), fraud detection (Abdallah et al., 2016), industrial fault and damage detection (Carrera et al., 2015), medical diagnosis (Schlegl et al., 2019), event data in earth sciences (Wu et al., 2019), and physics (Cerri et al., 2019). A notable example of AD is in the aerospace area where anomalous readings from a spacecraft sensor could signify a fault of components in the spacecraft (Fujimaki et al., 2005). AD is also related to novelty detection, which aims to detect previously unobserved patterns in the data, with the difference being that novel patterns are usually incorporated into the normal model after detection (Chandola et al., 2009). Despite many ADMs being introduced, AD has been considered a challenging topic, due to large variability within datasets and the lack of anomalous events for training (Ruff et al., 2021).

It is noted that many methods based on ML have been introduced for various applications where ML refers to the use of algorithms that learn from data to perform human-level tasks such as recognition and understanding. In recent years, a new subfield of ML called Deep Learning (DL) has achieved state-of-the-art results in many areas. DL utilizes model architecture with processing layers to learn data representations with multiple abstraction levels (LeCun et al., 2015). With DL, it becomes possible to automatically learn relevant features, with exceptional success in comparison to traditional methods. DL is the driving force behind many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. DL-based approaches to AD have delivered improved results on complex datasets and renewed interest in this area, with a great variety of new methods being introduced (Ruff et al., 2021). To our knowledge, there has been no review of ML and anomaly detection methods (ADMs) for GED. There are also no existing papers where DL-based methods were used for the GED problem in the literature or where a broader comparison was made with the different ADMs. We believe that it is important to (i) raise awareness in the chemical engineering community of an important application of ML/DL to the GED problem and (ii) showcase how newer ML/DL approaches can be used and performed on a number of benchmark systems. This motivates us to conduct an extensive review of the ADMs and evaluate their performances when solving the problem of GED so as to explore their potential in practical GED applications.

Our contributions are as follows:

- It is recognized that there are already published systems that aim at creating diverse benchmarks for DR and GED (do Valle et al., 2018), however, they have been used effectively for conventional approaches without focusing on ML/DL as well as any comparisons of existing methods. To apply ADMs to the problem of GED, training data is required from which to exploit the pattern of gross errors. To our knowledge, there is a lack of training datasets for the problem of GED in the literature which prevents the application of ADMs to this problem. In this study, we generate a number of measurement datasets, including training and testing data, associated with 16 systems introduced in the literature.
- We conduct an extensive review of existing ADMs based on ML and DL. The main approaches in the literature will be delineated and examples for each approach will be provided. Our review also highlights the main challenges currently faced and potential research directions.
- We propose a learning system to detect GEs by using ADMs. Our model describes a data pipeline consisting of multiple sequential steps from data pre-processing to detection model training and deployment. Our study aims to introduce a pipeline on how to implement and deploy an AD-based framework for the GED problem, starting from the raw measurement data to the trained detection model.
- We train 19 ADMs on the training datasets associated with the 16 systems to generate detection models. These models are applied to test datasets to determine whether anomalies are present in them. The performance of ADM models is evaluated against a number of established performance metrics. We explore what novelty and performance improvement ADMs can bring to the GED problem as well as provide a guidance of choosing an ADM in specific situations.

The paper is organized as follows. In Section 2, methods for GE detection and identification for steady-state cases, along with ML and DL-based ADMs are introduced. In Section 3, the general model of a GED system using ADMs and all settings for the comparative study are described. Experimental results are presented in Section 4; here the results of the ADMs and several existing GED methods are compared when using datasets generated from 16 systems collected from the literature. Finally, our conclusions are presented in Section 5.

2. Background and literature review

2.1. Methods for gross error detection and identification

There are four notable requirements when designing any GED method (Jordache and Narashimhan, 1999) (***i) detect the presence of one or more GEs (the detection problem) (ii) identify and locate the single GE (the identification problem) (iii) identify and locate multiple gross errors (MGE) present in the system (the MGE identification problem) (iv) GE magnitude estimation problem.*** In this section, we will mention several existing GED methods for steady-state cases (Jiang et al., 2014).

It is assumed that random error present in any measurement follows a normal distribution with zero mean and known variance. Therefore, the normalised error which follows the standard normal distribution mostly falls inside a confidence interval at a given or chosen significance level. This statistical principle is the basis of statistical tests to detect GEs. The statistical tests are commonly based on hypothesis testing by choosing between the null hypothesis H0 “no gross error is present” and the alternative hypothesis H1 “one or more gross errors are present”. The null hypothesis is accepted or rejected by comparing a test statistic

to an appropriate critical value. One of the most widely used statistical tests is the multivariate ***global test*** (GT) (Reilly and Carpani, 1963) in which the test statistic is calculated based on the vector of constraint residuals and its covariance matrix. The ***constraint test*** or ***nodal test*** (NT) (Mah et al., 1976) works on each constraint residual separately by exploiting diagonal terms of the covariance matrix of constraint residuals. Crowe (1989) introduced the ***maximum power constraint test*** based on a linear transformation of constraint residuals raising the probability of detecting GEs above that of the NT. The ***measurement test*** (Crowe and Garcia Campos, 1983) (MT) meanwhile treats each measurement separately when detecting GEs on each stream. MT uses measurement adjustments which are the differences between measurements and associated reconciled estimates to calculate the test statistic. Since MT or NT includes a set of univariate tests for streams or nodes and each test uses the same critical value, the probability of Type I error of one of the univariate tests in MT and NT will be higher than the specified value of the significance level. To overcome this issue, Mah and Tamhane (1982) proposed to modify the significance level based on the Sidak inequality so as to control the Type I error probability. Rollins and Davis (1992) proposed another modification of the significance level based on the Bonferroni confidence interval. Narasimhan and Mah (1987) introduced the ***Generalized Likelihood Ratio*** (GLR) test which aims to maximise a ratio of posterior probabilities of obtaining residual vectors under the H0 and H1 hypotheses to find GEs. Another approach which is used for gross error detection is Inter Quartile Range (IQR), in which an instance outside of the range between the first and third quartile is considered anomalous (Laurikkala et al., 2000). Tong and Crowe (1995) mentioned the limitations of several multivariate and univariate tests including GT, MT, and NT relating to ignorance of the covariance between each pair of elements in the covariance matrices. They then proposed principal component tests which provide more advantages by exploiting the entire information of the covariance matrix of constraint residuals and measurement adjustments through Principal Component Analysis (PCA)-based projections. Each of the above-mentioned methods aims to solve the detection problem. The NT, MT, GLR, and PCA tests also aim to identify and locate GEs while the GLR test can also estimate the GE value.

For the MGE identification problem, several methods have been introduced with one of three main strategies: ***(i) serial elimination (ii) serial compensation, and (iii) collective (or simultaneous) compensation*** (Prata et al., 2010).

Describing the ***serial elimination strategy***, Rosenberg et al. (1987) reviewed the early work in Ripps (1965), Nogita (1972), and Romagnoli and Stephanopolous (1981) such that when a GE is detected by using a GED method like GT or MT, the measurements are then removed sequentially in a group of specific size until no suspect set of measurements can be found. Yang et al. (1995) used the serial elimination strategy in designing a combined test from MT and NT. MT is applied to discover a potential stream list containing the GEs and then NT is applied on nodes associated with streams in that list to obtain the stream with the highest test statistic. This stream will be removed from the measurements and the iteration is run until no GEs are found. Congli et al. (2006) changed the order of the MT-NT combined method by applying NT first to detect potential nodes containing GEs and then applying MT on streams associated with these nodes. Jiang et al. (2014) applied GT and serial elimination strategy together to detect and identify GE for operational data in power plants. The detection results of their method were then validated by checking on-site inspection and maintenance records in order to ensure a reliable detection outcome.

In the ***serial compensation strategy***, GEs are also detected but instead of removing them from the measurement as in the elimination strategy, they are replaced by estimated values. Wang et al. (2004) applied this strategy to modify the combined MT-NT of (Yang et al., 1995) so that the coefficient matrix remained unchanged, reducing early termination of the combined test.

Finally, ***collective compensation strategy*** methods attempt to detect all GEs simultaneously in a single iteration. Keller et al. (1994) used the magnitudes estimated collectively by GLR (Narasimhan and Mah, 1987) to compensate for the GEs in case GEs are present in the systems. Sánchez et al. (1999) introduced a two-stage algorithm to simultaneously estimate biases and leaks in process plants. In the first stage, starting with each constraint, data reconciliation is applied to determine the associated objective function. The global test is run in each case by using the value of the objective function. If the null hypothesis H0 of this test is rejected, all measurements involved in the considered constraint and a leak from the corresponding node are added to the list of suspected gross errors. In the second stage, each combination including several candidates in the suspected list is evaluated. The combinations contain GEs if they raise the lowest objective function value and satisfy the global test i.e., H0 hypothesis is accepted. Loyola-Fuentes and Smith (2019) used the GT for GED and modified the simultaneous work of (Reilly and Carpani, 1963) with a non-linear programming implementation in order to detect GE in crude oil pre-heat trains undergoing shell-side and tube-side fouling deposition.

The importance of using historical information relating to past failure data on measuring instruments for further enhancement of test procedure performance emerged in the late 1990s. The Bayesian approach proposed by Tamhane et al. (1988) was the first attempt to employ such historical information to model the occurrences of GEs. Under distribution assumptions for the random error and binary indicator of GE on each measurement, the authors utilised the Bayesian method to model the posterior probability of the indicator given historical information. According to Bayes' rule, the GE indicator was determined by maximizing the posterior probability. Yuan et al. (2015) proposed a hierarchical Bayesian framework to solve both GED and DR problems. The measurement model was given in the same way as in Tamhane et al. (1988) concerning the GE indicator and magnitude, however, the posterior probability was proposed in a more complicated form by estimating the indicator and magnitude of GE, the reconciled measurement, and covariance matrices given by historical data. The complicated posterior probability was broken down into three inference layers for reconciled measurement and GE magnitude (layer 1), covariance matrices (layer 2), and GE indicator (layer 3) under Bayes rule. In recent years, with a proliferation of applications of ML in many areas, several ML-based models trained on historical data were introduced to detect and identify GEs. Reddy and Mavrovouniotis (1998) used a 3-layer neural network including one input layer, one hidden layer, and one output layer, to estimate the value of each measurement and its associated residual error. For each test sample, if the sum of the squares of the residuals does not fall within the established confidence limits, this sample is highly likely to contain a GE. Gerber et al. (2014) treated GED as a classification problem in which the ground truth of each measurement is encoded into binary value for detection and categorical value for the identification task. The authors experimented with 3 classifiers namely decision tree, linear, and quadratic discriminative analysis on 10 datasets generated from a simple two-product splitter process. Nguyen et al. (2020) proposed an ensemble of GED methods in which the outputs of each method given in the form of p-values are combined by using the Fisher combination method for the final GED conclusion. The proposed ensemble was further improved by searching for a suitable subset of methods in the ensemble for each sample. The authors used Particle Swarm Optimization (PSO), an effective swarm-based continuous optimization method, for the search process. Dobos et al. (2021) also introduced an ensemble of GED methods in which the outputs of all methods in the ensemble are combined by using a weighted combining method i.e., each method is associated with a specific weighted value in the combining.

2.2. Anomaly detection methods

2.2.1. Traditional machine learning methods for anomaly detection

ML refers to a class of algorithms that learn from data to perform

human-level tasks with reasonable accuracy. In the past years, although many ML methods have been introduced to detect anomalous samples from the data, there exists a number of challenges that make the design of an effective ML method for AD difficult. Firstly, the amount of labelled anomalous data is lacking, which has an adverse impact on the training of ML models. Secondly, each domain has a different notion of anomaly, which makes it difficult to transfer a technique developed in one domain to another. Thirdly, data often contains noise that is very hard to distinguish from the real anomalies, and this negatively affects the performance of the ML model. Fourthly, in many areas, the concept of anomaly evolves over time and the ML model must be updated accordingly. In this section, we will review the major approaches in applying ML to AD.

ML techniques for AD can be divided into several broad categories: density estimation and probability-based methods, one-class classification methods, reconstruction-based methods, and proximity-based methods. Density-based ADMs are based on the principle in which outliers are usually in low-density regions as opposed to normal instances (Li et al., 2022). One of the simplest methods in this class is the "three-sigma" rule, which considers points more than three standard deviations from the mean to be anomalies (Bakar et al., 2006). Classic density estimation methods for AD include kernel density estimator (Kim and Scott, 2012), Gaussian Mixture Model (GMM) (Amruthnath and Gupta, 2018), and histogram estimator (Ruff et al., 2021). It should be noted that the number of required parameters for these models increase exponentially as the number of dimensions increases. In Li et al. (2022), Zheng et al. proposed ECOD, which first finds the empirical cumulative distribution for each data dimension, then estimates the tail probabilities per dimension. Finally, an outlier score is calculated by aggregating estimated tail probabilities across dimensions. COPOD, proposed by Li et al. (2020), is built on the concept of copulas, which are functions that can separate marginal distributions of a given multivariate distribution, which allows separate modelling of each dimension. An empirical copula is first constructed which is then used to predict tail probabilities of each data point. Pevny (2016) proposed LODA, which uses a number of one-dimensional histograms constructed using random projection to approximate the joint probability, followed by an ensemble of histogram detectors. In Goldstein and Dengel (2012) the authors introduced HBOS, which calculates histograms using dynamic bin width for each dimension, and the product of the inverse of the estimated density is used as the final anomaly score. We can see a difference between HBOS and ECOD and COPOD in which while HBOS constructs histograms using random projection to approximate the joint probability, and the density in each histogram is used as an anomaly score, ECOD and COPOD model the distribution of each dimension separately before estimating the tail probabilities for the outlier detection. Although the histogram-based techniques are relatively simple, they may struggle to capture the interactions between different attributes. It is noted that density-based ADMs work based on statistical assumptions regarding the data they are applied to. Therefore, their performance is strongly influenced by the robustness of these assumptions. Those methods often provide confidence intervals along with their outputs, ensuring statistically justifiable results with guaranteed margins of error, as long as the underlying density and probabilistic assumptions are satisfied (Chandola et al., 2009).

One-class ADMs find an optimal decision boundary to differentiate the abnormal instances by learning from normal instances only during training so that the trained model can identify anomalous cases it has never seen before (Chandola et al., 2009). The objective of one-class classification is to learn the decision boundary using unlabelled data which minimizes the falsely raised alarms for normal instances (type I error) and undetected true anomalies (type II error) (Ruff et al., 2021) These methods are based on the argument that full density estimation is not necessary for AD, since only one single density level set is needed (Tax and Duin, 2004). One-class classification can be seen as binary classification in which only accessing normal data is available. The one-class classification objective is then to minimize the falsely raised alarms for normal instances and missed anomalies (Ruff et al., 2021). In this category, one-class Support Vector Machine (OCSVM) (Wang et al., 2007) is a popular technique for unsupervised AD. OCSVM works on the basic idea of minimizing the hypersphere of the single class of examples in training data and considers all the other samples outside the hypersphere to be outliers or out of training data distribution. Another notable method is Support Vector Data Description (SVDD) (Tax and Duin, 2004) which is based on the observation that a good data description covers all the target data but no superfluous space. A spherically shaped boundary is created around the dataset using a few training instances to separate the normal and abnormal instances. Elliptic Envelope, proposed in Hardin and Rocke (2004) seeks to find the subset of training instances whose covariance matrix has the lowest determinant (Rousseeuw and Driessen, 1999), and the constructed ellipse which covers these instances is then used to determine whether a point is anomalous or not. It is noted that one-class ADMs trained a model on normal instances only to create a decision boundary so that the model can identify anomalous cases it has never seen before. One-class ADMs do not put any statistical assumptions about the data distribution like other approaches such as density and probability-based, which make one-class ADMs more generalised than their peers (Ruff et al., 2021). On the other hand, one-class ADMs return binary output on each test instance, which can also be seen as a disadvantage when a meaningful anomaly score in the form of probability is desired for decision-making (Chandola et al., 2009).

Reconstruction-based methods for AD meanwhile learn a model which can reconstruct normal instances while failing at reconstructing anomalous instances (Ruff et al., 2021). Most of these models, such as Principal Component Analysis (PCA) (Shyu et al., 2003), are motivated by geometrical considerations, but some are connected to density estimation. There are two assumptions underlying these methods: the manifold and prototype assumptions. The manifold assumption asserts that the data is mostly represented by some lower dimension manifold which resides in the data space, while the prototype assumption asserts that there exists a finite number of prototypical elements in the data space which reasonably characterizes the data. One of the most popular reconstruction-based methods is autoencoders, which is a neural network consisting of two parts: encoder and decoder (Ruff et al., 2018). The encoder embeds the input data into a lower-dimensional space while the decoder reconstructs the resulting embedding into the original space. The autoencoder is then trained to minimize the reconstruction error between the input and the output of the decoder, which forces the network to learn the best representation of the data in the hidden layer. The data instances with high reconstruction errors are assumed to be anomalies. Another example is Variational Autoencoder (VAE) which adopts a stochastic autoencoding process by encoding and decoding the parameters through the encoder and decoder network but unlike autoencoders, VAE assumes that the latent representation has Gaussian distribution, and the VAE is trained to minimize the mean reconstruction errors (Kingma and Welling, 2019). In Shyu et al. (2003), the authors proposed to use PCA for anomaly detection based on the observation that the first few principal components explain most of the variance in the data and large values of the last components represent significant deviations from the normal instances. The principal components are extracted, and an instance is classified as anomalous if the sum of the first or last components exceeds a certain threshold. Kriegel et al. (2009) proposed SOD which detects anomalies using the deviations from the neighbours of each instance in the axis-parallel subspace spanned by the neighbours. Among these methods, autoencoders and VAE project the data into a lower dimension before projecting back to the original space by using a neural network. The anomalies would have higher reconstruction errors compared to normal instances. Other methods such as PCA meanwhile use principal components of the data for reconstruction. Reconstruction-based ADMs are capable of handling high-dimensional datasets due to their ability to perform dimensionality reduction. They

can also serve as a pre-processing step for input data before applying a different method in another category (Chandola et al., 2009). However, it should be noted that certain reconstruction-based methods, such as PCA and autoencoder, are not suitable for datasets with high levels of noise as they may yield poor results (Chalapathy et al., 2017).

Proximity-based ADMs separate anomalous instances from normal instances using a predefined proximity metric. In proximity-based methods, an instance is anomalous if its locality, or proximity, is sparsely populated. There are three ways of defining proximity: cluster-based, distance-based, and density-based (Aggarwal, 2017). In the cluster-based approach, the dataset is divided into a number of clusters and an instance is classified as anomalous depending on its non-membership in the clusters, its distance from the nearest cluster, or the size of the closest cluster. In the distance-based approach, the distance from an instance to its nearest neighbours is used to define proximity, while in the density-based approach, the number of other points in a local region is used to define which instance is anomalous. In Liu et al. (2009), the authors proposed Isolation Forest, which is based on the observation that tree structure can be constructed to isolate anomalous instances at the root. An ensemble of these isolation trees is created to detect the anomalous instances which have shorter average path lengths than the normal instances. Local Outlier Factor (LOF), proposed in Breunig et al. (2000), computes the local density deviation of a given data point compared to its neighbours. The instances with a significantly lower density than their neighbours are considered anomalous. Angiulli and Pizzuti (2002) proposed to use the sum of distances from an instance to its k-nearest neighbours as the anomaly scores, which are efficiently calculated by linearizing the search space. In Kriegel et al. (2008), the authors noted that approaches based on distance calculation are not effective on high-dimensional data due to the curse of dimensionality. They proposed Angle-Based Outlier Detection (ABOD) which uses the variance of the angles between a point and all other pairs of points in the dataset as the anomaly score. Almardeny et al. (2020) proposed Rotation-based Outlier Detection (ROD), which first decomposes the full attribute space into different subspaces, then rotates the data instances about the geometric mean to construct the anomaly score. In Tang et al. (2002) the authors noted that LOF implicitly assumes that the data is distributed spherically, which might be restrictive. The authors proposed a Connectivity-based Outlier Factor (COF) method, based on the concept of chaining distance, which is the minimum of the sum of the distance of the k neighbours and the instance. The authors noted that chaining distance can be used as an approximation for local density of the neighbourhood while the ratio of the chain distance of a data point to the average chain distance of all nearest neighbours at that point is used to define the anomaly score. In Arning et al. (1996), the authors proposed Linear Method for Deviation Detection (LMDD), a linear-complexity algorithm which minimises dissimilarity between instances to detect the anomalies using a dissimilarity function and a smoothing factor. Among these methods, density-based methods such as LOF uses the number of other points in a local region as a criterion to detect anomalies. In contrast, distance-based methods such as COF and LMDD use the distance from an instance to its nearest neighbours as a proximity measure, while cluster-based methods like ROD separates the data into a number of clusters so that an instance is classified based on its membership values in the clusters. Proximity-based ADMs have an advantage from the fact that they do not require any statistical assumptions about the data, and thus that makes them suitable for many types of datasets. By contrast, there are some disadvantages of proximity-based ADMs in which their performances are largely dependent on the proximity measure to distinguish between normal and anomalous instances, the computation requirements for the testing phase might be high, and if the data has either normal instances without enough neighbours or abnormal instances with enough neighbours then there will be missed anomalies.

2.2.2. Deep learning methods for anomaly detection

DL is an emerging subfield of ML which utilizes model architecture with processing layers to learn data representations with multiple abstraction levels (LeCun et al., 2015). With DL, it becomes possible to automatically learn relevant features, with exceptional success over traditional methods (Hinton et al., 2012), especially in computer vision (Krizhevsky et al., 2012). DL-based approaches to AD have delivered improved state-of-the-art results on complex datasets and renewed interest in this area, with a great variety of new methods being introduced (Ruff et al., 2021). Unlike traditional methods, DL-based approaches to AD mitigate the burden of feature engineering and enables effective, scalable solutions. The main approaches in applying DL to AD are: deep autoencoders variant, deep one-class classification, deep generative models, such as Generative Adversarial Networks (GAN), variants, and self-supervised methods (Ruff et al., 2021).

An autoencoder consists of two parts: the encoder and the decoder. The encoder maps the input to a smaller dimension to extract salient features, and the decoder maps these features back to the original dimension to reconstruct the input (Ruff et al., 2021). Some popular variants of deep autoencoders for AD include denoising autoencoders (Zhou and Paffenroth, 2017), variational autoencoders (VAE) (Ruff et al., 2021), Adversarial Autoencoders (Principi et al., 2017), and Recurrent Neural Network (RNN)-based autoencoders (Lu et al., 2017). An example of Adversarial Autoencoder in AD is SO-GAAL (Liu et al., 2020), which uses a mini-max game between a generator and a discriminator to generate informative potential. The generator directly generates artificial anomalies which are closed to real data through the guidance of the discriminator. Artificial anomalies enable the discriminator to learn to distinguish between normal instances and anomalies based on a separation boundary between them. In Chalapathy et al. (2017), the authors proposed an improvement to Robust PCA (Candes et al., 2010) by replacing linear projection with a deep and robust autoencoder, which allows the method to capture complex non-linear structures for AD. In Aytekin et al. (2018), showed that adding L2 normalization for the training of deep autoencoders results in more separable clusters. Based on this observation, the authors performed k-mean clustering on the resulting L2 normalized representation of the deep autoencoder, and the distances from the centroids to each sample can be used for AD. In Gong et al. (2019), the authors noted the assumption that anomalies always correspond to high reconstruction error might not always be correct and used a memory module (Weston et al., 2015) to augment the autoencoder. Given an input, the method firsts obtain an encoding from the encoder, which is then used to query the most relevant items in the memory for reconstruction. In the training stage, the memory content is encouraged to represent the normal data. Liu et al. (2021) proposed a robust framework based on collaborative autoencoders to jointly identify normal observations in the dataset while learning its feature representation for AD.

One-class Support Vector Machine (OCSVM) (Wang et al., 2007) is one of the most popular one-class classification techniques for AD. However, it is known that the complexity of SVM grows quadratically with the number of training instances, which is a serious challenge for high-dimensional datasets (Huang and LeCun, 2006). In Erfani et al. (2016), the authors proposed to use Deep Belief Network (DBN) for feature extraction before applying OCSVM for AD. Ruff et al. (2018) observed that most deep ADMs involve networks trained on a separate task which is then adapted to AD. The authors proposed Deep SVDD, which uses a deep neural network jointly trained to map the data into a hypersphere of minimal volume. Deep SVDD does not suffer from two limitations of OCSVM and the original SVDD relating to quadratically scaling with the number of samples and large amount of memory requirement to store the support vectors. An important problem usually encountered in deep one-class classification is feature map collapse without regularization (Ruff et al., 2021). Possible solutions for this problem include regularization (Wu et al., 2020), freezing the embeddings (Erfani et al., 2016), or using pseudo-labelling (Golan and

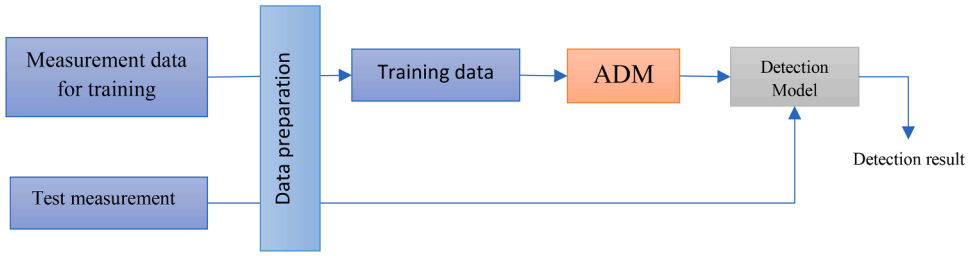


Fig. 1. The model of a gross error detection system using machine learning-based anomaly detection methods.

El-Yaniv, 2018). A variant of one-class SVM for time-series is introduced in (Shen et al., 2020) in which a dilated recurrent neural network with skip connections is used to capture temporal connections. A one-class objective for time series is then obtained by using multiple hyperspheres in a hierarchical clustering process. Oza and Patel (2019) proposed a one-class Convolutional Neural Network (CNN), in which a pre-trained CNN is trained using pseudo-negative class created from zero-centered Gaussian noise in the latent space. Unlike other approaches, this method allows the use of transfer learning (Weiss et al., 2016) for AD.

Deep generative-based ADMs use deep generative models such as GAN to generate artificial anomalies and a discriminator will learn to differentiate between an anomaly and a normal instance. A GAN consists of a generator and a discriminator network in which these two networks are adversarially trained with the generator trained to deceive the discriminator, and the discriminator trained to differentiate between the input and the output by the generator Ruff et al. (2021). In Perera et al. (2019), the authors proposed OCGAN, which exclusively constrains the latent space to represent the given class. An adversarial discriminator is used in the input space. A GAN-inspired method is proposed in Sabokrou et al. (2018), in which two deep networks compete for AD. One network performs novelty detection, while the other enhances the normal samples and distorts the outliers. The method proposed in Pidhorskyi et al. (2018) followed a probabilistic approach and computed the probability that a sample was generated by the inlier distribution. This is achieved by linearizing the parameterized manifold capturing the underlying inlier distribution and factorizing the probability calculation using local coordinates of the manifold tangent space. It should be noted that GAN suffers from significant problems relating to training stability (Ruff et al., 2021). Zenati et al. (2018) proposed Adversarially Learned Anomaly Detection (ALAD), which used bi-directional GAN to adversarially derive learned features for AD while ensuring GAN stability during training, which leads to significantly improved performance. In Akcay et al. (2018), the authors introduced GANomaly, which used a conditional GAN to jointly model the generation of high-dimensional image space and the inference of latent space. Encoder-decoder-encoder subnetworks allowed the mapping of the input image to its latent representation, and the distance between the generated output image and the latent vectors are minimized. Among these methods, OCGAN is trained similarly to the original GAN, while other methods use different types of GAN such as bi-directional GAN, conditional GAN, and encoder-decoder-encoder subnetworks (e.g. GANomaly) to adversarially derive learned features for AD. It is noted that the instances generated by using deep generative ADMs models such as GAN are shown to be realistic compared to the real data. With the involvement of generated instances in the adversarial training methods, the deep generative ADMs models might obtain good performance on some kinds of datasets. The drawback meanwhile is that these models are trained using an alternating optimization scheme, which usually does not provide stable outputs i.e. different runs may yield different outputs (Ruff et al., 2021).

Self-supervised learning-based ADMs use a predefined auxiliary task to help the model distinguish between anomalies and normal instances without using labelled data (Ruff et al., 2021; Zhang et al., 2022). A wide range of auxiliary tasks has been proposed in self-supervised learning, such as colorization, rotation, cropping, or masked word prediction. Recent studies have also demonstrated that the representations learned from self-supervised tasks can improve AD performance, provided that the anomaly score and the auxiliary task are chosen correctly (Hojjati et al., 2022). There are two types of self-supervised learning for AD: self-predictive methods and contrastive methods. In self-predictive methods, a transformation is applied to the input sample and the model either predicts the applied transformation or reconstructs the original input. An example of self-predictive methods is CutPaste (Li et al., 2021), which notes that geometric transformations such as rotation and translation are effective in learning semantic concepts but not regularity. In CutPaste, small random rectangular regions are cut and pasted at another location. A deep network is then trained to distinguish normal images from these images. Another example is the method in Zhang et al., (2022) which used an auxiliary classification task in which normal examples are combined with instances sampled from several distributions, such as Gaussian and Poisson, for the method to learn the normal data features. In contrastive methods, the objective of the auxiliary task is to emphasize the contrast between “positive” and ”negative” samples to make the models learn more effectively. In Tack et al. (2020), the authors proposed Contrasting Shifted instances (CSI), which seeks to contrast a given example not only with other instances but also distributionally shifted augmentations of itself. It should be noted that despite many successes in multiple areas, until recently self-supervised learning AD has been mostly applied to image data and other types of data in which the auxiliary task can be easily defined (Hojjati et al., 2022), which might not be the case for many domains. Self-supervised ADMs have a significant advantage in which the pretext tasks which these methods use do not require ground truth labels when training the detection model. This makes them suitable for a wider range of datasets where obtaining ground truth information can be costly or unavailable (Ruff et al., 2021). On the other hand, the appropriate pretext task needs to be carefully chosen to obtain a good performance. For example, Hojjati et al. (2022) suggested that some methods based on pretext tasks like geometric transformations and contrastive methods work well for detecting semantic anomalies, whereas other methods based on pixel-level transformations are more appropriate for defect detection. Since it is difficult to define the auxiliary task for GED, in this paper we do not include self-supervised ADMs in our experiments.

3. Experimental studies

Fig. 1 shows how a ML-based AD model for GED works. Measurement data for training are first processed, for example, missing values are removed, or some measurements are transformed and normalized before becoming the training data. A ML-based ADM is then trained on the training data to generate a detection model. Among ML paradigms with different settings, supervised and unsupervised ML are the most popular ones. In supervised learning, we have a labelled dataset $\mathbf{D}$ including N observations $(\mathbf{x}, y_{\mathbf{x}})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ is a feature vector that belongs to the input domain, and $y_{\mathbf{x}}$ is the class label of $\mathbf{x}$ which belongs to the target domain. The relationship between $\mathbf{x}$ and $y_{\mathbf{x}}$ can be described by an unknown function f i.e., $y_{\mathbf{x}} = f(\mathbf{x})$. Supervised

Table 1
The main approaches in anomaly detection using machine learning and deep learning.

Deep Learning (DL) based, or Machine Learning (ML) based	Approach	References
ML	Reconstruction-based models	Autoencoders (Ruff et al., 2018), PCA (Shyu et al., 2003), SOD (Kriegel et al., 2009)
	One-class classification models	One-class SVM (Wang et al., 2007), SVDD (Tax and Duin, 2004), Elliptic Envelope (Hardin and Rocke, 2004)
	Density estimation and probability models	GMM (Amruthnath and Gupta, 2018), Kernel density estimator (Kim and Scott, 2012), Histogram estimator (Ruff et al., 2021), ECOD (Li et al., 2022), COPOD (Li et al., 2020), LODA (Pevny, 2016), HBOS (Goldstein and Dengel, 2012)
	Proximity-based models	Isolation Forest (Liu et al., 2009), LOF (Breunig et al., 2000), (Angiulli and Pizzuti, 2002), ABOD (Kriegel et al., 2008), (Almardeny et al., 2020), COF (Tang et al., 2002), KNN (Ruff et al., 2021), LMDD (Arning et al., 1996), ROD (Almardeny et al., 2020)
DL	Reconstruction-based models	Denoising autoencoders (Zhou and Paffenroth, 2017), Variational autoencoders (VAE) (Ruff et al., 2021), Robust autoencoder (Chalapathy et al., 2017), L2-normalized deep autoencoder (Aytekin et al., 2018), Memory-augmented deep autoencoder (Gong et al., 2019), Collaborative deep autoencoder (Liu et al., 2021), Adversarial Autoencoder (Principi et al., 2017), RNN-based autoencoder (Lu et al., 2017), SO-GAAL (Liu et al., 2020)
	One-class classification models	Hybrid DBN-1SVM (Erfani et al., 2016), Deep SVDD (Ruff et al., 2018), RNN-based OCSVM (Shen et al., 2020), One-class CNN (Oza and Patel, 2019), Robust Deep PCA (Chalapathy et al., 2017)
	Deep generative models	OCGAN (Perera et al., 2019), (Sabokrou et al., 2018), (Pidhorskyi et al., 2018), ALAD (Zenati et al., 2018), GANomaly (Akcay et al., 2018)
	Self-supervised models	CSI (Tack et al., 2020), (Zhang et al., 2022), CutPaste (Li et al., 2021)

ML algorithms aim to propose an approximation (also called hypothesis) g for the function f. On that basis, we apply g to predict the class label of unseen samples. On the other hand, the class label y_x is not given in unsupervised learning so the learning algorithm focuses on exploring hidden knowledge and patterns of the data from N feature vectors only. For the GED problem, when the GE information of measurement data is given, we can train a supervised ML algorithm on the training data. Several methods mentioned in Section 2 like the Bayesian approach (Yuan et al., 2015; Dobos et al., 2021) and Neural Network (Reddy and Mavrovouniotis, 1998) used this ML paradigm to train a GED system. However, to our knowledge and experience, the requirement concerning available GE information is hard to fulfil in real-life applications because of an expensive checking inspection on the historical measurement data. The setting of unsupervised ML meanwhile seems to be more realistic and feasible for the problem of GED. The details of each component and process of the proposed model in Fig. 1 are given as follows:

Data preparation: Due to the heterogeneous origin and setup of data acquisition devices, real-world data may be collected with either redundant, incomplete, or inconsistent information. Applying ML algorithms to that raw data may not obtain high-quality results as they would fail to identify patterns effectively. Therefore, collected measurement data may need to be prepared before it can be used as training data for an ADM. Data preparation is the process of sorting and filtering the raw data to remove unnecessary and inaccurate data to make it in a suitable form for further analysis and processing. There is no unique set of tasks in the data preparation process, however, the process for GED systems typically involves several main tasks including (i) data structuring: the data is modelled and organized to meet the analytics requirements (ii) data cleansing: missing and inconsistent information on the data is imputed and resolved (iii) data selection: a subset of redundant readings was selected (iv) data transformation: the data is transformed into a unified and usable format.

Data preparation was found in several existing studies concerning the GED problem including the data transformation in the PCA test of Tong and Crowe (1995) and data selection and transformation in the PCA and Neural Network-based method of Reddy and Mavrovouniotis (1998). PCA serves as the foundation for multivariate data analysis using projection methods. It is frequently used to obtain lower-dimensional data while retaining most of the data's variation by projecting each data point onto only the first few principal components. PCA is important in terms of representing multivariate data in a smaller set of variables which can improve the observation of trends, clusters, and outliers in the data. The measurement selection procedure in redundant readings was also mentioned in Reddy and Mavrovouniotis (1998) in which 44 variables were selected among 65 variables of readings. The selected variables were then transformed by three sequential steps: replacing exhibited strong linear correlation by their first principal component score, mean-centering, and scaling into a specific range.

In this study, we use PCA and Random Projection on measurement data for the data transformation. Like PCA, Random Projection (Johnson and Lindenstrauss, 1984; Nguyen et al., 2019) linear transformation in which the pairwise distance between pairs of points can be preserved with a specific probability and distortion level ε before and after conducting a projection. Random projections are useful in dimension reduction if the dimension of the projected data is chosen to be lower than that of the original data. Random projections are simply obtained by using a $p \times q$ random matrix $\mathbf{R} = \{r_{ij}\}$ where p and q are the dimensions of the original data and projected data, respectively) and the $\{r_{ij}\}$ are random variables such that $\mathbb{E}(r_{ij}) = 0,\ \mathrm{Var}(r_{ij}) = 1$. Several forms of $\{r_{ij}\}$ are given by Nguyen et al. (2019):

- Plus-minus-one or Bernoulli random projection: $\mathbf{R} = 1/\sqrt{q}\{r_{ij}\}$ where r_{ij} is randomly chosen from $\{-1,\ 1\}$ such that $\Pr(r_{ij} = 1) = \Pr(r_{ij} = -1) = 1/2$.
- Achlioptas random projection: $\mathbf{R} = 1/\sqrt{q}\{r_{ij}\}$ where r_{ij} is randomly chosen from $\{-\sqrt{3},\ 0,\ \sqrt{3}\}$ such that $\Pr(r_{ij} = \sqrt{3}) = \Pr(r_{ij} = -\sqrt{3}) = 1/6$ and $\Pr(r_{ij} = \sqrt{3}) = 2/3$.
- Normal random projection: $\mathbf{R} = 1/\sqrt{q}\{r_{ij}\}$ where r_{ij} is distributed according to $\mathcal{N}(0,\ 1)$ distribution.

Training data: The training data is the output of data preparation in a well-prepared instance to train a detection model. In the model in Fig. 1, the training dataset is fed to an ADM to teach it how to detect the presence of GEs. In this study, we generated training data on 16 systems collected from the literature (please find these systems' information in Fig. S5-S20 in the Supplement Material). Our study covered a large range of systems, starting from only 3 streams up to 50 streams. All the selected systems were introduced in the benchmark paper from the GED problem (do Valle et al., 2018). Table S.9. in the Supplement Material shows the number of streams of the benchmark scenarios, and the characteristics of the streams (parallel stream, recycled stream, or

Table 2
Anomaly detection models used in the experiments and their hyperparameter settings.

Model name	Library	Hyper-parameters
Isolation Forest (denoted by iForest)	Scikit-learn	Number of trees: 500.
Local Outlier Factor (denoted by LOF)	Scikit-learn	Number of neighbours: 5.
One-class SVM (denoted by OCSVM)	Scikit-learn	RBF kernel. Nu: 0.5.
Elliptic Envelope	Scikit-learn	Proportion of outliers in the dataset: 0.1.
KNN	PyOD	Number of neighbours: 5.
ECOD	PyOD	Proportion of outliers in the dataset: 0.1.
COPOD	PyOD	Proportion of outliers in the dataset: 0.1.
ABOD	PyOD	Number of neighbours: 5.
ROD	PyOD	Proportion of outliers in the dataset: 0.1.
LODA	PyOD	Number of histogram bins: 10. Number of random cuts: 100.
Autoencoder	PyOD	Number of hidden neurons per layer: (Krizhevsky et al., 2012; Congli et al., 2006; Congli et al., 2006; Krizhevsky et al., 2012). Activation function: ReLU. Number of epochs: 100. Optimiser: Adam.
VAE	PyOD	Number of neurons in encoder: [128], (Krizhevsky et al., 2012; Congli et al., 2006). Number of neurons in decoder: (Congli et al., 2006; Krizhevsky et al., 2012),[128]. Number of epochs: 100. Optimiser: Adam.
SO-GAAL	PyOD	Discriminator learning rate: 0.01. Generator learning rate: 0.0001. Stop epoch: 20. Optimiser: SGD.
Deep SVDD	PyOD	Number of neurons per hidden layer: (Krizhevsky et al., 2012; Congli et al., 2006). Activation function: ReLU. Number of epochs: 100. Optimiser: Adam.
PCA	PyOD	Proportion of outliers in the dataset: 0.1.
LMDD	PyOD	Number of iterations: 50.
COF	PyOD	Number of neighbours: 20.
HBOS	PyOD	Number of bins: 10.
SOD	PyOD	Number of neighbours: 20.

measurement) that can be used as a reference when selecting the right datasets for the experiments. In these systems, the true values of measurements and the variances of random errors associated with all streams were given. First, the random error was generated under the assumption of a normal distribution with zero mean and given variance. The generated random errors for all streams were added to the true measurements to create the base case data i.e., no-GEs data. Then GEs were generated for all streams and added to those base cases. In this work, each GE was generated under a uniform distribution between −25% to +25% of the associated measurement value. The training data for a system of m streams contains 1000 samples with non-GE, 10 samples with GE on the m^{th} stream. Hence, the training data includes $m*10+1000$ samples $\mathbf{x} = (x_1, x_2, \ldots, x_D)$. The ground truth of GE presentation is not required for the training since we experiment on unsupervised ADMs. The details of the systems in the experiments can be found in the Supplemental Material.

ADM and Detection Model: To train ADMs effectively, it is necessary to have properly initialized hyper-parameters. Each method has a different set of hyper-parameters that normally affect its performance. In DL-based methods, the number of neurons in each layer and the number of layers are two important hyper-parameters to specify the architecture of the DL system. The optimiser which would find the optimal weights of neurons in the network can yield very different detection results according to different choices, although recent optimisers such as Adam have improved the optimisation process considerably. The learning rate of the optimiser is also an important hyper-parameter since small learning rates might make the computational time increase while large learning rates might make the model oscillate. Finally, the number of epochs, which is the number of times a DL-based method passes through the dataset during training, needs to be determined carefully to reduce underfitting or overfitting situations during the training. Meanwhile, the parameters of ML-based methods are usually approach-dependent. For example, the proximity approach such as LOF and KNN requires the number of neighbours from which the anomalies can be determined. Tree-based methods like Isolation Forest require a number of trees from which to create an ensemble of trees for collaborated detection. The density-based methods like LODA require the number of histogram bins and the number of random cuts to determine the density of the training data.

In the experiments, we used 19 ML or DL-based ADMs from 4 different categories of ADMs (please see Table 1) to ensure that the selected methods cover all approaches to handle anomalous instances. The selected methods have been introduced recently or are state-of-the-art models for AD (Chandola et al., 2009) and have not yet been applied to the problem of GED. Four ML-based methods namely Isolation Forest (Liu et al., 2009), LOF (Breunig et al., 2000), One-class SVM (Wang et al., 2007), and Elliptic Envelope (Hardin and Rocke, 2004) are implemented by using the Scikit-learn library (Pedregosa, 2011), while the remaining methods are implemented by using the PyOD library (Zhao et al., 2019) in which three models, VAE (Ruff et al., 2021), SO-GAAL (Liu et al., 2020), and Deep SVDD (Ruff et al., 2018) are DL-based while the remaining are ML-based methods. Among the methods implemented in the Scikit-learn library, two are one-class classification models (One-class SVM and Elliptic Envelope) while the other two are proximity-based methods (Isolation Forest and LOF). Meanwhile, among the methods implemented in the PyOD library, five methods are proximity-based (ABOD (Kriegel et al., 2008), COF (Tang et al., 2002), KNN (Ruff et al., 2021), LMDD (Arning et al., 1996), ROD (Almardeny et al., 2020)), four methods are density estimation and probability-based (ECOD (Li et al., 2022), COPOD (Li et al., 2020), LODA (Pevny, 2016), HBOS (Goldstein and Dengel, 2012)), five methods are reconstruction-based (Autoencoder (Ruff et al., 2018), VAE (Ruff et al., 2021), PCA (Shyu et al., 2003), SOD (Kriegel et al., 2009), SO-GAAL (Liu et al., 2020)), and one method is one-class classification method (Deep SVDD (Ruff et al., 2018)).

In the experiment, we used the values of the hyperparameters provided by the authors of the original papers or ML/DL libraries, which in our opinion can give a good basis for comparison for two reasons. First, practitioners may face challenges to use some models arising from areas in which they are not experts because of the requirements on the knowledge and experience for the tuning parameter procedure. While it may be possible through tuning to achieve better performances on specific datasets, we aim to demonstrate that competitive results to the benchmark algorithms (statistical tests) can be obtained with the predefined hyperparameters, obviating the need for an additional tuning overhead for each application. By using the default or suggested hyperparameters' value for the models, we can provide a more achievable result that would fit most applications. Besides, although tuning the hyperparameters of experimental methods can improve GE detection results, it normally would take longer if we would try to showcase the methods with different hyperparameters, especially for DL-based models. The detailed hyper-parameter settings of each method in the experiment are given in Table 2.

Testing process: In the model in Fig. 1, a test sample is first passed through the data preparation process. All preparation steps applied to measurement data for training once again will be applied to each test sample in the same order. The data preparation step outputs the suitable format to input to the detection model. In this study, test samples were also generated on 16 systems to study the performance of ADMs. We generated two test datasets for each system in which each GE was generated under a uniform distribution between −5% and +5% of the associated measurement value for the first one, and between −10% and +10% of the associated measurement value for the second one. It is recognised that the first test dataset is more challenging than the second one for the GED task because of the smaller magnitudes of GEs. The data

Table 3
The Accuracy of the 19 experimental ADMs on the 16 datasets with ±5% of GEs.

	iForest	LOF	OCSVM	Elliptic envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO-GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.499	0.454	0.687	0.491	0.497	0.323	0.360	0.463	0.311	0.394	0.454	0.472	0.407	0.289	0.458	0.451	0.287	0.462	0.284
P2	0.430	0.527	0.788	0.572	0.561	0.224	0.215	0.535	0.213	0.361	0.488	0.521	0.292	0.163	0.497	0.392	0.171	0.393	0.182
P3	0.421	0.553	0.756	0.804	0.517	0.162	0.120	0.505	0.160	0.246	0.746	0.784	0.077	0.152	0.778	0.156	0.250	0.406	0.113
P4	0.475	0.631	0.838	0.646	0.637	0.215	0.211	0.620	0.222	0.476	0.586	0.609	0.143	0.159	0.561	0.406	0.172	0.395	0.186
P5	0.307	0.357	0.729	0.368	0.376	0.211	0.207	0.348	0.195	0.189	0.309	0.337	0.111	0.145	0.352	0.473	0.132	0.303	0.139
P6	0.213	0.205	0.514	0.209	0.210	0.206	0.208	0.212	0.199	0.204	0.206	0.206	0.194	0.204	0.206	0.406	0.161	0.212	0.125
P7	0.243	0.358	0.683	0.357	0.381	0.190	0.189	0.337	0.197	0.168	0.292	0.303	0.091	0.166	0.313	0.327	0.287	0.233	0.126
P8	0.259	0.211	0.917	0.246	0.232	0.187	0.180	0.201	0.163	0.195	0.219	0.243	0.083	0.166	0.221	0.467	0.292	0.219	0.119
P9	0.214	0.196	0.923	0.255	0.174	0.162	0.173	0.193	0.166	0.171	0.209	0.226	0.077	0.155	0.216	0.277	0.274	0.223	0.195
P10	0.248	0.354	0.685	0.351	0.344	0.152	0.123	0.338	0.137	0.146	0.202	0.228	0.197	0.099	0.210	0.362	0.100	0.289	0.100
P11	0.199	0.174	0.923	0.184	0.217	0.173	0.166	0.199	0.152	0.186	0.169	0.184	0.077	0.135	0.187	0.243	0.253	0.211	0.153
P12	0.265	0.573	0.786	0.635	0.569	0.173	0.157	0.606	0.158	0.177	0.525	0.593	0.077	0.228	0.605	0.317	0.115	0.293	0.110
P13	0.181	0.093	0.578	0.167	0.178	0.152	0.152	0.145	0.148	0.171	0.159	0.159	0.158	0.205	0.165	0.275	0.079	0.183	0.075
P14	0.328	0.066	0.947	0.256	0.291	0.151	0.119	0.105	0.149	0.151	0.080	0.085	0.040	0.137	0.083	0.397	0.185	0.216	0.183
P15	0.248	0.052	0.691	0.082	0.164	0.157	0.140	0.063	0.098	0.148	0.068	0.071	0.034	0.212	0.070	0.163	0.195	0.165	0.172
P16	0.355	0.032	0.980	0.033	0.077	0.166	0.151	0.031	0.139	0.134	0.020	0.020	0.020	0.147	0.020	0.334	0.091	0.217	0.033
Ave	0.305	0.302	**0.777**	0.354	0.339	0.188	0.179	0.306	0.175	0.220	0.296	0.315	0.130	0.173	0.309	0.340	0.190	0.276	0.143

Table 4
The Accuracy of the 19 experimental ADMs on the 16 datasets with ±10% of GEs.

	iForest	LOF	OCSVM	Elliptic envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO-GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.866	0.791	0.849	0.861	0.867	0.323	0.344	0.847	0.321	0.650	0.805	0.830	0.505	0.533	0.821	0.476	0.298	0.809	0.307
P2	0.814	0.847	0.902	0.885	0.878	0.221	0.199	0.845	0.195	0.300	0.886	0.896	0.294	0.372	0.897	0.475	0.180	0.540	0.211
P3	0.670	0.779	0.845	0.997	0.775	0.165	0.117	0.737	0.173	0.290	0.998	0.998	0.166	0.459	0.998	0.314	0.152	0.505	0.109
P4	0.887	0.880	0.926	0.916	0.907	0.213	0.192	0.880	0.188	0.887	0.930	0.940	0.143	0.235	0.927	0.485	0.170	0.479	0.193
P5	0.628	0.570	0.888	0.738	0.688	0.197	0.185	0.613	0.187	0.396	0.726	0.756	0.111	0.216	0.763	0.450	0.147	0.411	0.172
P6	0.245	0.232	0.573	0.258	0.246	0.212	0.213	0.255	0.208	0.257	0.248	0.259	0.201	0.206	0.248	0.372	0.186	0.239	0.143
P7	0.365	0.550	0.760	0.569	0.569	0.177	0.169	0.533	0.200	0.314	0.560	0.566	0.222	0.218	0.563	0.365	0.188	0.272	0.135
P8	0.524	0.398	0.917	0.638	0.446	0.172	0.150	0.396	0.152	0.452	0.581	0.626	0.083	0.184	0.592	0.343	0.266	0.293	0.150
P9	0.349	0.388	0.923	0.514	0.360	0.162	0.151	0.377	0.173	0.263	0.499	0.520	0.077	0.185	0.505	0.178	0.273	0.305	0.156
P10	0.456	0.741	0.871	0.765	0.689	0.156	0.119	0.595	0.158	0.503	0.767	0.787	0.173	0.144	0.773	0.297	0.103	0.470	0.108
P11	0.372	0.303	0.923	0.558	0.366	0.150	0.129	0.354	0.146	0.286	0.513	0.550	0.077	0.198	0.563	0.219	0.142	0.316	0.121
P12	0.459	0.802	0.882	0.920	0.767	0.164	0.145	0.788	0.159	0.368	0.949	0.959	0.320	0.644	0.956	0.363	0.095	0.364	0.128
P13	0.266	0.095	0.658	0.279	0.280	0.148	0.128	0.213	0.144	0.263	0.270	0.273	0.197	0.191	0.277	0.166	0.129	0.245	0.135
P14	0.482	0.067	0.952	0.762	0.634	0.148	0.109	0.303	0.135	0.538	0.753	0.758	0.040	0.333	0.754	0.263	0.077	0.264	0.075
P15	0.368	0.055	0.813	0.366	0.406	0.151	0.120	0.157	0.109	0.241	0.336	0.347	0.034	0.258	0.344	0.285	0.171	0.280	0.219
P16	0.525	0.037	0.980	0.039	0.278	0.172	0.154	0.057	0.143	0.128	0.020	0.020	0.020	0.222	0.020	0.405	0.049	0.308	0.037
Ave	0.517	0.471	0.854	0.629	0.572	0.183	0.164	0.497	0.174	0.384	0.615	0.630	0.166	0.287	0.625	0.341	0.164	0.381	0.150

Table 5
The OP of the 19 experimental ADMs on the 16 datasets with ±5% of GEs.

	iForest	LOF	OCSVM	Elliptic Envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO-GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.365	0.307	0.739	0.349	0.363	0.116	0.153	0.314	0.097	0.218	0.303	0.326	0.243	0.083	0.307	0.336	0.064	0.311	0.056
P2	0.345	0.460	0.830	0.511	0.500	0.101	0.085	0.468	0.084	0.270	0.413	0.450	0.190	0.036	0.423	0.313	0.035	0.309	0.047
P3	0.377	0.519	0.773	0.790	0.483	0.095	0.047	0.467	0.091	0.185	0.726	0.767	0.000	0.091	0.762	0.086	0.187	0.361	0.039
P4	0.394	0.578	0.880	0.594	0.583	0.090	0.080	0.564	0.094	0.396	0.524	0.550	0.000	0.034	0.494	0.315	0.038	0.304	0.052
P5	0.227	0.284	0.757	0.295	0.305	0.119	0.109	0.274	0.098	0.097	0.228	0.259	0.000	0.048	0.278	0.439	0.023	0.227	0.031
P6	0.115	0.105	0.514	0.109	0.110	0.106	0.108	0.113	0.098	0.103	0.106	0.107	0.091	0.106	0.105	0.370	0.044	0.114	0.001
P7	0.177	0.301	0.698	0.300	0.326	0.116	0.112	0.277	0.123	0.092	0.229	0.242	0.000	0.096	0.253	0.271	0.220	0.166	0.039
P8	0.200	0.147	1.000	0.184	0.169	0.119	0.108	0.135	0.091	0.128	0.153	0.180	0.000	0.102	0.155	0.449	0.229	0.157	0.039
P9	0.154	0.136	1.000	0.197	0.111	0.096	0.107	0.131	0.101	0.107	0.146	0.165	0.000	0.096	0.154	0.234	0.214	0.167	0.128
P10	0.193	0.306	0.697	0.302	0.296	0.088	0.056	0.289	0.072	0.085	0.141	0.169	0.142	0.037	0.150	0.328	0.031	0.240	0.031
P11	0.137	0.112	1.000	0.119	0.159	0.108	0.099	0.139	0.085	0.125	0.103	0.119	0.000	0.073	0.122	0.190	0.192	0.151	0.083
P12	0.207	0.540	0.807	0.606	0.535	0.107	0.088	0.577	0.089	0.115	0.487	0.561	0.000	0.172	0.575	0.264	0.041	0.240	0.036
P13	0.136	0.039	0.585	0.120	0.133	0.105	0.102	0.096	0.099	0.125	0.111	0.111	0.111	0.166	0.118	0.241	0.021	0.139	0.017
P14	0.304	0.028	0.984	0.226	0.263	0.118	0.083	0.068	0.115	0.118	0.042	0.047	0.000	0.108	0.045	0.382	0.151	0.187	0.149
P15	0.226	0.019	0.700	0.050	0.137	0.129	0.111	0.030	0.067	0.120	0.035	0.038	0.000	0.192	0.037	0.136	0.167	0.138	0.143
P16	0.345	0.013	1.000	0.014	0.059	0.151	0.135	0.012	0.123	0.119	0.000	0.000	0.000	0.134	0.000	0.324	0.073	0.203	0.013
Ave	0.244	0.243	0.810	0.298	0.283	0.110	0.099	0.247	0.095	0.150	0.234	0.256	0.049	0.098	0.249	0.292	0.108	0.213	0.057

Table 6
The OP of the 19 experimental ADMs on the 16 datasets with ±10% of GEs.

	iForest	LOF	OCSVM	Elliptic Envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO-GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.858	0.754	0.967	0.849	0.858	0.114	0.126	0.826	0.096	0.565	0.773	0.810	0.377	0.408	0.798	0.344	0.075	0.779	0.103
P2	0.794	0.834	0.966	0.877	0.869	0.098	0.066	0.831	0.061	0.199	0.877	0.889	0.190	0.279	0.890	0.399	0.050	0.479	0.090
P3	0.647	0.763	0.870	1.000	0.762	0.099	0.044	0.719	0.105	0.236	1.000	1.000	0.101	0.427	1.000	0.258	0.082	0.468	0.035
P4	0.876	0.869	0.988	0.910	0.898	0.088	0.058	0.867	0.053	0.875	0.924	0.935	0.000	0.127	0.921	0.400	0.037	0.403	0.067
P5	0.590	0.523	0.935	0.712	0.656	0.101	0.083	0.572	0.086	0.329	0.698	0.732	0.000	0.131	0.739	0.391	0.042	0.350	0.069
P6	0.152	0.136	0.582	0.165	0.153	0.111	0.107	0.162	0.106	0.165	0.155	0.167	0.096	0.106	0.153	0.318	0.070	0.144	0.021
P7	0.309	0.510	0.785	0.534	0.534	0.098	0.086	0.492	0.122	0.254	0.522	0.530	0.155	0.152	0.525	0.305	0.109	0.209	0.049
P8	0.489	0.350	1.000	0.612	0.402	0.101	0.073	0.346	0.076	0.411	0.548	0.599	0.000	0.122	0.561	0.293	0.203	0.239	0.073
P9	0.302	0.344	1.000	0.479	0.314	0.096	0.082	0.332	0.107	0.212	0.463	0.486	0.000	0.127	0.470	0.116	0.213	0.258	0.086
P10	0.418	0.723	0.896	0.748	0.667	0.093	0.052	0.568	0.095	0.468	0.750	0.771	0.117	0.088	0.757	0.245	0.034	0.435	0.039
P11	0.325	0.251	1.000	0.524	0.320	0.082	0.057	0.307	0.076	0.232	0.475	0.515	0.000	0.141	0.530	0.157	0.070	0.265	0.048
P12	0.418	0.788	0.912	0.915	0.750	0.097	0.074	0.775	0.089	0.323	0.946	0.958	0.270	0.623	0.955	0.310	0.020	0.318	0.055
P13	0.226	0.042	0.669	0.239	0.241	0.098	0.074	0.168	0.093	0.222	0.230	0.232	0.155	0.153	0.236	0.117	0.074	0.204	0.081
P14	0.464	0.030	0.988	0.752	0.620	0.115	0.072	0.274	0.100	0.522	0.742	0.748	0.000	0.313	0.744	0.234	0.039	0.237	0.037
P15	0.350	0.022	0.827	0.344	0.387	0.122	0.089	0.127	0.078	0.216	0.313	0.324	0.000	0.241	0.321	0.263	0.141	0.258	0.191
P16	0.519	0.018	1.000	0.020	0.264	0.157	0.138	0.039	0.127	0.113	0.000	0.000	0.000	0.211	0.000	0.398	0.030	0.296	0.018
Ave	0.484	0.435	0.899	0.605	0.543	0.104	0.080	0.463	0.092	0.334	0.589	0.606	0.091	0.228	0.600	0.284	0.081	0.334	0.066

Table 7
The Friedman test results for 19 methods on 32 datasets.

Metric	Chi-squared	df	p-value
Accuracy	331.74	18	< 2.2e-16
F1	331.41	18	< 2.2e-16
OP	333.29	18	< 2.2e-16
Selectivity	344.8	18	< 2.2e-16

generation for testing data is the same as the training data generation. According to a system with m streams, we generated the test data of $(m+1)*1000$ samples in which 1000 samples with no-GE, 1000 samples with GE on the m^{th} stream. The information details of experimental datasets generated from 16 systems are shown in Table A1.

The performances of all ADMs on the test samples were reported according to 4 performance metrics namely Overall power (OP), Selectivity, Accuracy, and F1-Score. OP and Selectivity are two popular metrics when evaluating the performance of GED methods while Accuracy and F1-Score are two popular metrics used in the ML community to evaluate the performance of ML methods. We illustrate the confusion matrix which summarises how successful an ADM is at predicting samples belonging to GE or non-GE class. Four performance metrics are computed based on the confusion matrix by using the Eqs. (1)–(4). Accuracy aims to measure all the correctly identified cases i.e., considering both true positive and true negative cases of a GE detector. The OP is intuitively the ability of a GE detector to find all the positive samples i.e., GE samples. The Selectivity meanwhile is intuitively the ability of a GE detector not to label as positive a sample that is negative i.e., non-GE. It is recognised that the OP is the Recall while the Selectivity is the Precision, two popular performance matrices used in the ML community. The F1 score takes into consideration both False Negative and False Positive in the detection results as the metric is the harmonic mean of the Precision (Selectivity) and Recall (OP).

Actual GE	Predicted GE	GE = Yes	GE = No
	GE = Yes	True Positive (TP)	False Negative (FN)
	GE = No	False Positive (FP)	True Negative (TN)

$$\text{Accuracy} = \frac{\text{Number of samples correctly detected}}{\text{Number of simulation samples made}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

$$\text{OP} = \text{Recall} = \frac{\text{Number of gross errors correctly identified}}{\text{Number of gross errors simulated}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$\text{Selectivity} = \text{Precision} = \frac{\text{Number of gross errors correctly identified}}{\text{Total number of gross errors detected}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

We used the Friedman test (Garcia and Herrera, 2008) to test the difference between the performances concerning each of these 4 metrics of 19 experimental ADMs on 32 test datasets. The Friedman test is used to test the null hypothesis that all methods perform equally on the datasets. If the null hypothesis is rejected i.e., the p-value of the Friedman test is smaller than a specific threshold, a post-hoc test is then conducted. In this case, we used Nemenyi post-hoc test for all pairwise comparisons based on the rankings of ADMs on all experimental test datasets. Two methods are considered to perform differently with a statistical significance if the p-value computed from the post-hoc test statistic is smaller than an adjusted value of confidence level computed from Nemenyi's procedure. In this work, the confidence level of the Nemenyi test was set to 0.05.

4. Experimental results and discussions

4.1. Comparison among ADMs

Tables 3-6 show the accuracy and OP results of the 19 experimental ADMs on the 32 test datasets. The F1 Score and Selectivity results of the ADMs can be found in the Supplemental Material. The Friedman test returns p-values smaller than 2.2E-16 for each performance metric which means we reject the null hypothesis of "no difference in the performances of all ADMs" (Table 7). The Nemenyi post-hoc test results in Figs.2-5 show a similar pattern relating to the accuracy, F1 score, and OP in which OCSVM ranks first among 19 methods on these performance metrics.

4.1.1. Comparison based on detection accuracy

For accuracy, the Nemenyi post-hoc test result indicates that OCSVM ranks first among all experimental methods and is better than 15 ADMs (PCA, iForest, LMDD, Autoencoder, ABOD, HBOS, LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL). Elliptic Envelop ranks second and is better than 9 ADMs (LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL). The top 5 ADMs ranked based on accuracy are OCSVM, Elliptic Envelope, KNN, VAE, and PCA in which the Nemenyi test indicates that there are no differences in the performance of OCSVM, Elliptic Envelope, KNN, and VAE. The 5 poorest methods based on accuracy are COF, ROD, COPOD, SOD, and SO-GAAL.

In detail, OCSVM obtains an average accuracy of 0.777 on 5% GE datasets, which is higher than the second-best method (Elliptic Envelope) by more than 40%. There are 9 ADMs (iForest, LOF, Elliptic Envelope, KNN, ABOD, Autoencoder, VAE, PCA, and LMDD) which obtain accuracies between 30% and 35%. SO-GAAL and SOD are the two poorest methods in this experiment in which their average accuracies are only about 14%. It is observed that P16 is the most challenging dataset for the experimental methods in which 10 methods obtained the poorest results on this dataset compared to those on the other datasets. Meanwhile, on the P1 dataset, 8 methods obtained the highest value of accuracy among their results on 16 datasets.

On 10% GE datasets, the accuracies of 13 methods increase at different rates while the performances of the others remain the same or even poorer than their performances on 5% GE datasets. Autoencoder and PCA are two methods with the most significant increase of 0.319 and 0.316. By contrast, COF, COPOD, and ECOD show a decrease in their accuracies (0.190 to 0.164 of COF, 0.179 to 0.164 of COPOD, and 0.188 to 0.183 of ECOD). Even though the average accuracy of OCSVM increases by only 0.077, this method is still the best ADM on 10% GE datasets (0.854 vs. 0.630 of the second-rank method VAE).

On 5% GE cases, OCSVM ranks first on 15 datasets. This method is outperformed by Elliptic Envelope, VAE, and PCA on the P3 datasets, obtaining 5% smaller accuracy than the first ranked method (Elliptic Envelope). When the magnitude of GEs increases to 10%, the better performance of OCSVM is not as clear as in the 5% cases in which OCSVM ranks first on 12 datasets. On the P1 and P4 dataset, OCSVM ranks fourth while on P3 and P12 datasets, OCSVM ranks fifth among all methods. The differences between the accuracy of OCSVM and the top performance methods on the P1 and P5 dataset are not significant. On the P1 dataset, for example, the accuracy of OCSVM is 0.849, which is about 2% smaller than the top 3 methods on this dataset (0.861 of Elliptic Envelope, 0.866 of iForest, and 0.87 of KNN). On the P4 dataset, OCSVM obtained an accuracy of 0.926 which is slightly smaller than that of PCA (0.927), Autoencoder (0.930), and VAE (0.940). Meanwhile, on P3 and P12 datasets, although OCSVM performed well, it was outperformed by some ADMs, for example, 0.845 vs. 0.998 of Autoencoder, VAE, and PCA on the P3 dataset.

4.1.2. Comparison based on overall power

For OP, the null hypothesis that all methods performed similarly is rejected. The Nemenyi post-hoc test result in Fig. 3 shows that OCSVM

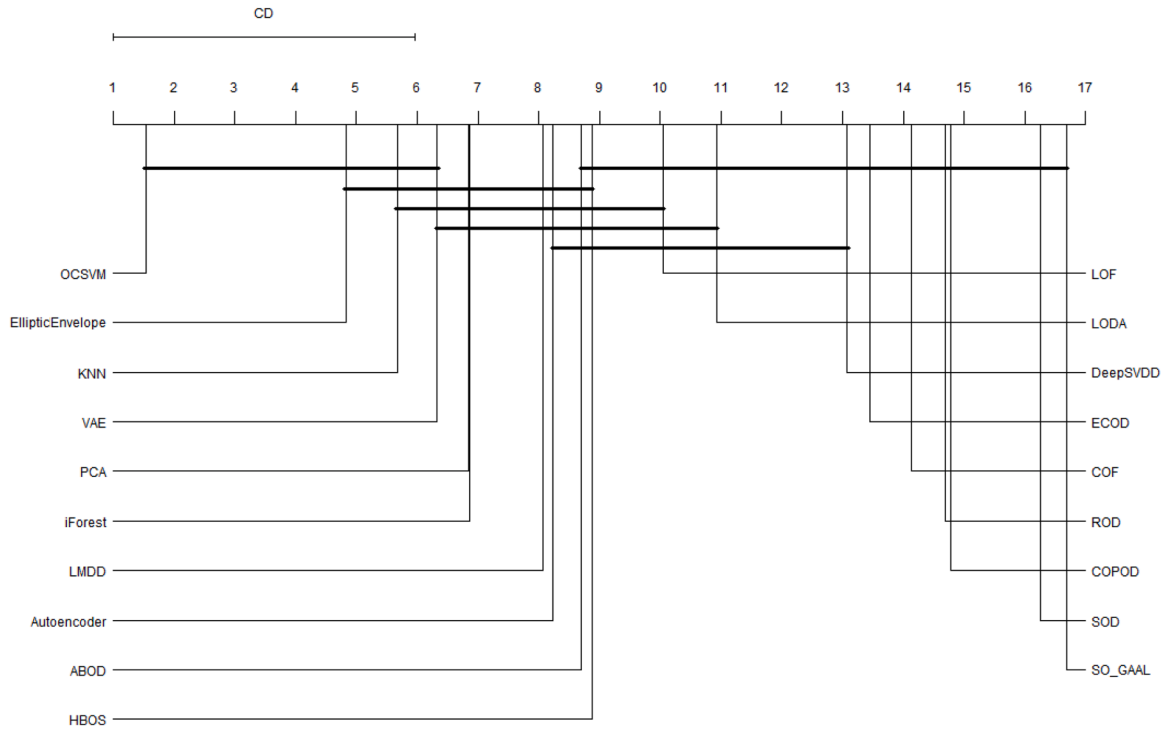


Fig. 2. Nemenyi test results on Accuracies of 19 ADMs
OCSVM > PCA, iForest, LMDD, Autoencoder, ABOD, HBOS, LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
Elliptic Envelope > LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
KNN > LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
VAE, PCA, iForest, LMDD > Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
Autoencoder > ECOD, COF, ROD, COPOD, SOD, SO-GAAL.

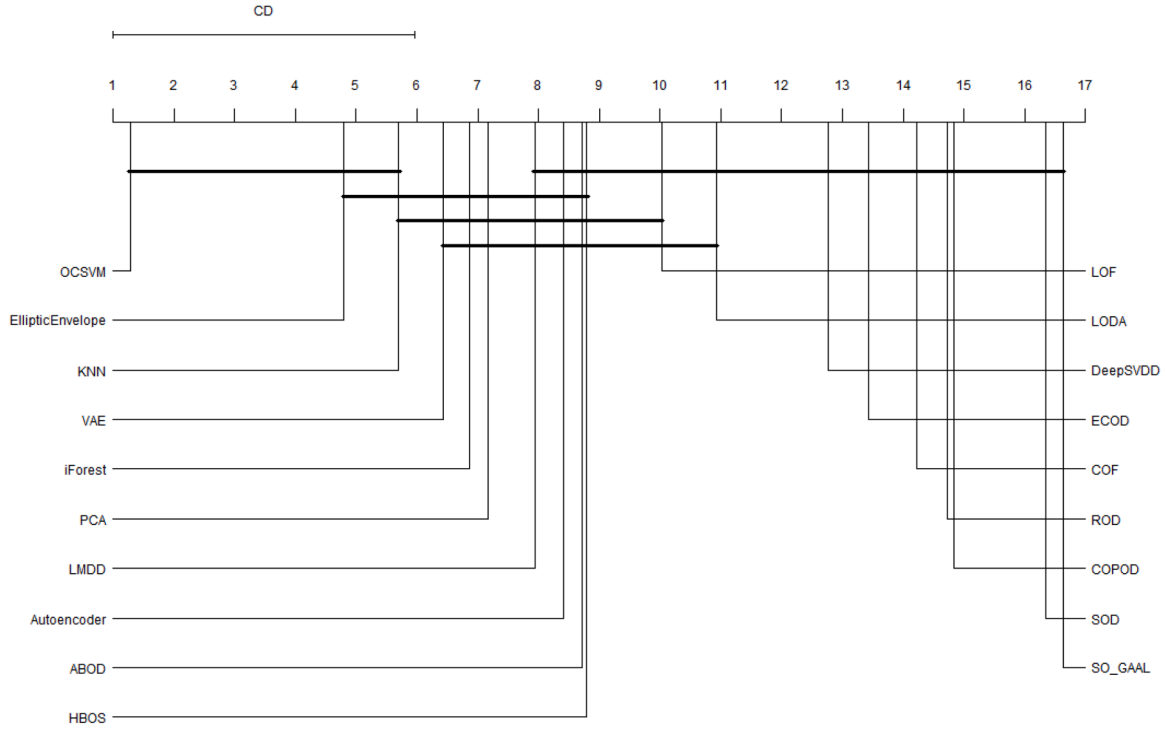


Fig. 3. Nemenyi test results on OPs of 19 ADMs
OCSVM > VAE, iForest, PCA, LMDD, Autoencoder, ABOD, HBOS, LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
Elliptic Envelope > LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
KNN > LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
VAE, PCA, iForest > Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL.

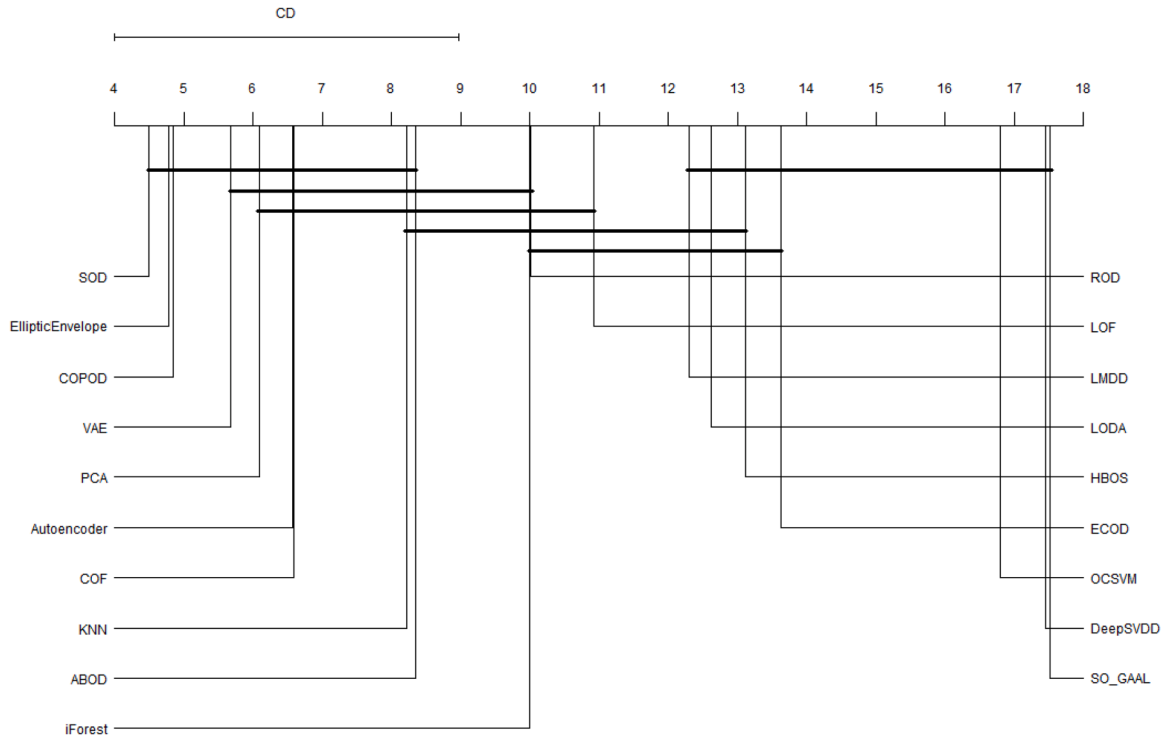


Fig. 4. Nemenyi test results on selectivity values of 19 ADMs
SOD, Elliptic Envelope, COPOD > iForest, ROD, LOF, LMDD, LODA, HBOS, ECOD, OCSVM, Deep SVDD, SO-GAAL
VAE > LOF, LMDD, LODA, HBOS, ECOD, OCSVM, Deep SVDD, SO-GAAL
PCA, Autoencoder, COF > LMDD, LODA, HBOS, ECOD, OCSVM, Deep SVDD, SO-GAAL
KNN, ABOD > ECOD, OCSVM, Deep SVDD, SO-GAAL
iForest, ROD, LOF > OCSVM, Deep SVDD, SO-GAAL.

obtains better results compared to the VAE, iForest, PCA, LMDD, Autoencoder, ABOD, HBOS, LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, and SO-GAAL on the 32 datasets. On 5% GE datasets, OCSVM ranks first on 15 datasets and ranks second on the P3 datasets. On 10% GE datasets, OCSVM ranks first on 14 datasets and ranks fifth on the P3 and P12 dataset. Compared to the results relating to accuracy where OCSVM ranks fourth on the P1 and P4 dataset for OP, OCSVM ranks first on these datasets. It is noted that OP is the Recall, which is intuitively the ability of a GE detector to find all the GE samples. This means OCSVM has a powerful ability to detect GE samples in a test dataset.

Once again, the Elliptic Envelope is the second-best method in our experiment. Although this method did not perform well on the 5% GE datasets like P6, P11, P15, and P16, on the 10% GE datasets, its OP result increases significantly on 13 datasets except P4 dataset (an increase from 0.880 to 0.910), P6 dataset (an increase from 0.109 to 0.165) and P16 dataset (an increase from 0.014 to 0.020). On P1 and P14 dataset, for example, the OPs of the Elliptic Envelope increase about 50% when GE magnitude increases from 5% to 10%. VAE performed in the same manner as Elliptic Envelope in which it did not obtain high performances on 5% GE datasets but improved the performances on 10% GE datasets. This method unfortunately could not detect any GE samples on P16 dataset (its OP is 0.000 which means its true positive is 0.000).

4.1.3. Comparison based on selectivity

The performances of all ADMs for Selectivity are shown in Table S1 and S2 in the Supplemental Material. We again reported the Nemenyi post-hoc test results in Fig. 4 for the pairwise comparison. On this performance metric, SOD, Elliptic Envelope, COPOD, VAE, and PCA are the top 5 methods. It is observed that 18 ADMs obtain high values of selectivity on the 5% GE datasets except for SO-GAAL on some datasets. On 10% GE datasets, the results of those 18 methods relating to the selectivity are even better than those on 5% GE datasets: many methods obtained Selectivity of 99%.

The Nemenyi test results indicate that SOD, Elliptic Envelope, and COPOD are better than iForest, ROD, LOF, LMDD, LODA, HBOS, ECOD, OCSVM, Deep SVDD, and SO-GAAL. It is noted that SOD is the second poorest method among 19 ADMs on the other performance metrics. The high value of the selectivity of SOD is from its small value of false positive because this method is only able to detect a very small number of GE samples and most of these detections are correct. A similar observation can be found in the case of COPOD.

In contrast, Elliptic Envelope and VAE are two methods obtaining the second and fourth ranks concerning the Selectivity metric. This demonstrates the ability of these methods not to label as positive a sample that is negative i.e., non-GE. OCSVM, meanwhile, was slightly outperformed by some ADMs concerning the Selectivity metric (OCSVM's average Selectivity score was 0.931 for up to 5% GEs compared to the best average Selectivity score was 0.978 of COPOD), although this method ranks first concerning the other performance metrics. That means OCSVM tends to predict non-GE samples to have GEs, resulting in slightly high values of the false positive (or slightly low values of the Selectivity).

4.1.4. Comparison based on F1 score

We compared the ADMs based on the F1 Score (please see the results in Table.S3 and Table.S4 in the Supplemental Material). In this case, the p-value obtained by using the Friedman test is smaller than 0.05. Hence, we rejected the null hypotheses and conducted the post-hoc test for all pairwise comparisons among the 19 methods. A similar pattern can be observed from the Nemenyi test results on accuracy and Overall Power. Based on the Nemenyi test, OCSVM is better than iForest, PCA, LMDD, Autoencoder, ABOD, HBOS, LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL. The second-ranked method Elliptic Envelope

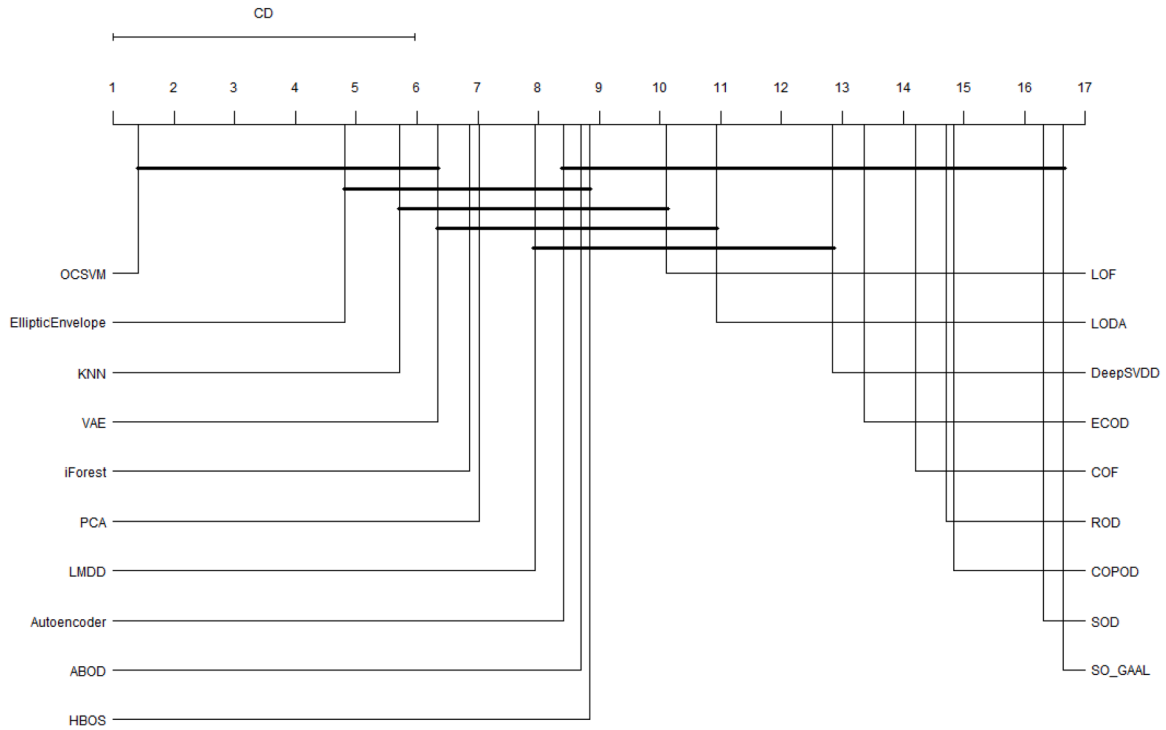


Fig. 5. Nemenyi test results on F1 scores of 19 ADMs
OCSVM > iForest, PCA, LMDD, Autoencoder, ABOD, HBOS, LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
Elliptic Envelope > LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
KNN > LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
VAE, PCA, iForest > Deep SVDD, ECOD, COF, ROD, COPOD, SOD, SO-GAAL
LMDD > ECOD, COF, ROD, COPOD, SOD, SO-GAAL.

meanwhile is better than LOF, LODA, Deep SVDD, ECOD, COF, ROD, COPOD, SOD, and SO-GAAL. Once again, OCSVM ranks first among 19 methods, followed by Elliptic Envelope and KNN. The top 5 poorest methods are COF, ROD, COPOD, SOD, and SO-GAAL.

On 5% GE datasets, OCSVM ranks first on up to 15 datasets for F1 score. This method performed poorly on the P3 dataset only, ranking fourth on this dataset and obtaining a 3% smaller value for the F1 score than the first-rank method. On 10% GE datasets, OCSVM ranked first on 13 datasets, ranked second on P4 dataset, and ranked fifth on P3 and P12 datasets. It is noted that OCSVM ranks fourth on P1 and P4 dataset with 10% GE for accuracy. The high values of F1 score of OCSVM indicate that this method obtained a good balance between the Selectivity (Precision) and OP (Recall) since F1 Score is the harmonic mean of those metrics.

To sum up, the experimental results and the statistical test results indicate that:

- OCSVM is the best ADM in the experiment concerning accuracy, F1 score, and OP. This method has a strong ability to detect GE samples (high values of OP or Recall) and achieves a balance between Precision (Selectivity) and Recall (OP) (high values of F1 score). OCSVM obtained slightly smaller values of selectivity compared to the other methods. This indicates that OCSVM tends to assign the positive label to non-GE samples slightly more than other methods.
- Elliptic Envelope is the second-best ADM for the GED in our experiment, maintaining the second rank for four performance metrics. Although this method did not perform well on 5% GE datasets, its performance improved significantly on 10% GE datasets. Elliptic Envelop underperformed compared to OCSVM in general but is much better than that competitor on the selectivity metrics. In practice, the Elliptic Envelope can be considered in datasets with large values of the magnitude of GEs.
- Other ADMs like KNN, PCA, iForest, and VAE performed in the same manner as Elliptic Envelope, performing poorly on 5% GE datasets but performing well on 10% GE datasets. These methods maintained the middle ranks concerning 4 performance metrics.
- SO-GAAL, SOD, COPOD, ROD, COF, ECOD, Deep SVDD, and LODA are the poorest methods to detect GE in our experiments. Although SOD obtained the best results relating to selectivity, this method can correctly detect a very small number of GE samples.

We put some explanations about the performance differences among experimental methods. It is noted that the 16 systems we used in the experiments contain linear relationships between streams (do Valle et al., 2018). As a result, the methods like OCSVM or Elliptic Envelope which provide decision boundaries between normal and abnormal instances with consideration to the linear relations on data attributes can obtain high performance on the experimental datasets. Besides, KNN and PCA obtained good results on 10% GE datasets but performed poorly on 5% GE datasets. This could be due to their sensitivity to certain parameters such as the proportion of outliers in the dataset. Previous research (Haakon et al., 2007) showed that PCA is highly sensitive to its parameters. Additionally, since PCA transforms data to a different space, anomalous instances may exhibit either better or worse discriminative characteristics when distinguished from normal instances. Therefore, while PCA may perform well on some datasets, it may perform poorly on others.

Poor-performing methods such as ECOD and COPOD were shown to be ineffective in detecting dependency anomalies in previous research (Songqiao et al., 2022) (ECOD and COPOD ranked 1st and 3rd as the poorest methods in their experiments to detect dependency anomalies). In our datasets, instances were generated according to process constraints (such as equations of mass or energy balance) between streams, resulting in dependent relationships between stream variances. This

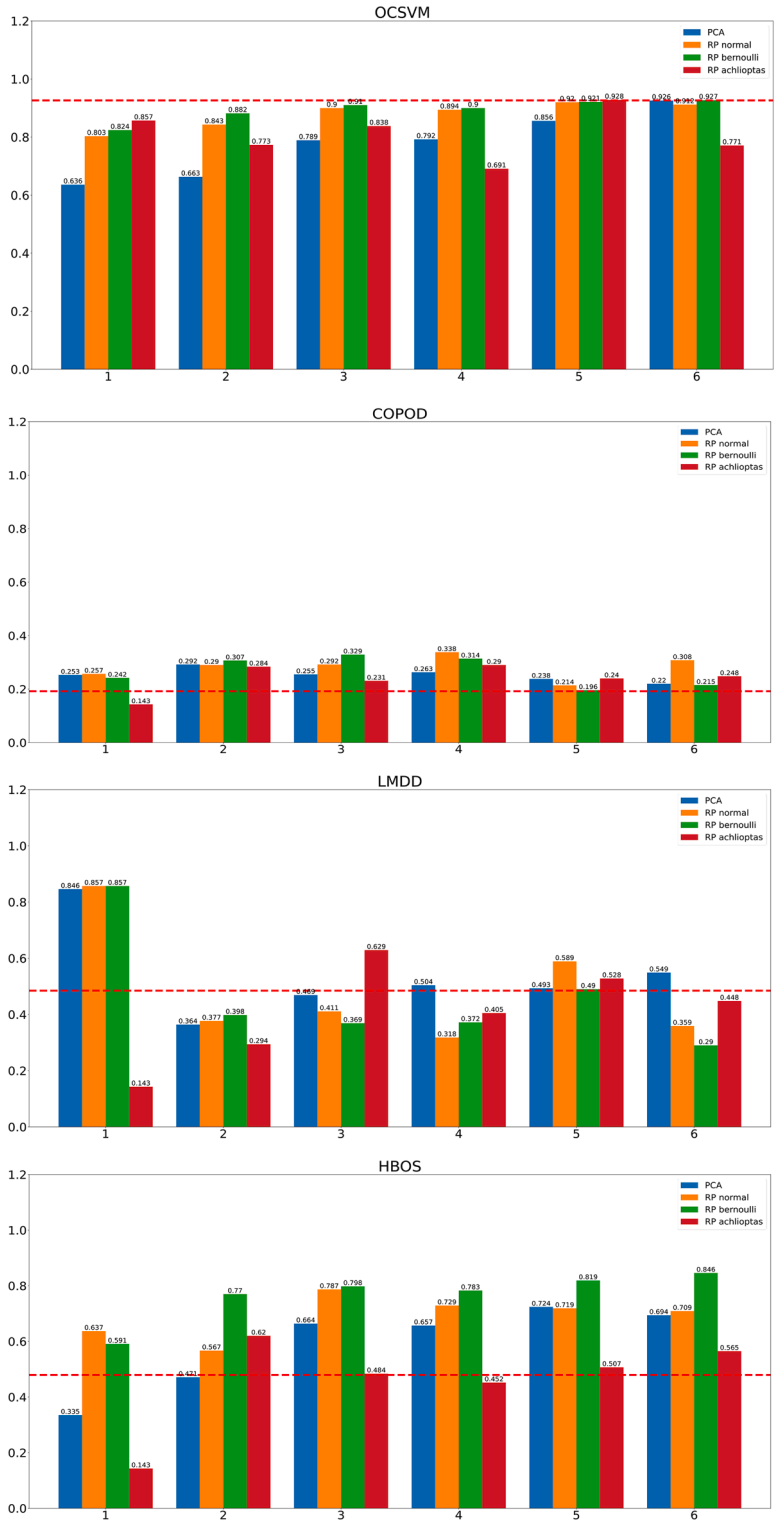


Fig. 6. The accuracies of 4 selected ADMs on the P4 dataset (10% GE) with different data transformation methods.

explains why ECOD and COPOD performed poorly on the experimental datasets.

4.2. The impacts of data preparation

We investigate the impacts of data preparation on the performance of ADMs on the experimental datasets. The training data was transformed by using PCA or RP before feeding into an ADM. Fig. 6 shows the Accuracy of 4 selected ADMs on the experimental systems with a different number of retained components or dimensions. The red dashed lines in these figures show the accuracy of an ADM on the original P4 dataset with 10% of GE. We denoted "Method Name+PCA", "Method Name+Normal", "Method Name+Bernoulli", and "Method Name-+Achlioptas" as the names of an ADM training on dataset transformed by using a data transformation method. The figures of the other methods as well as the figures relating to OP, F1-Score, and Selectivity outputs

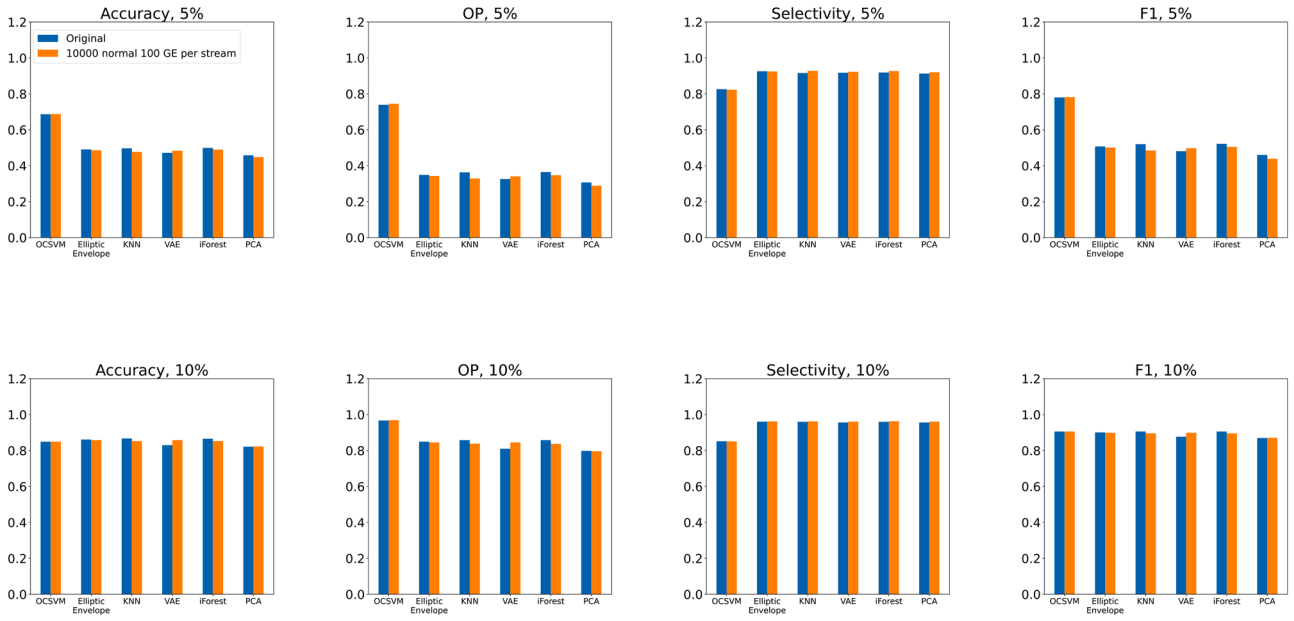


Fig. 7. The performance of 6 selected ADMs on P1 datasets.

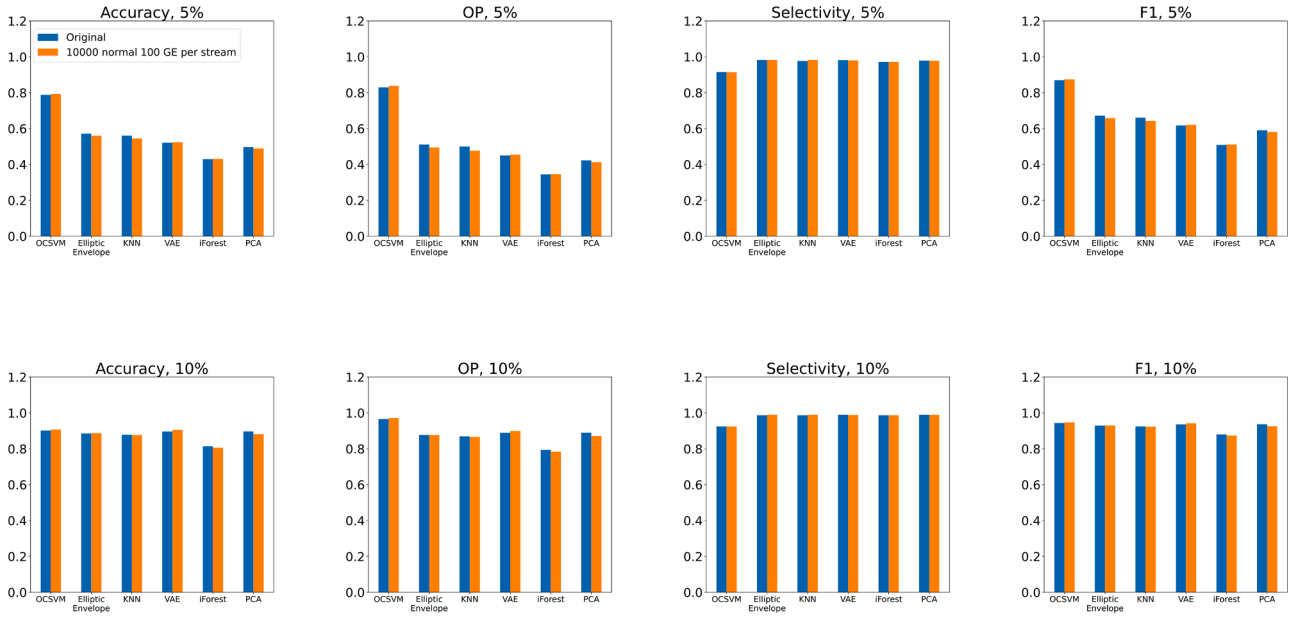


Fig. 8. The performance of 6 selected ADMs on P2 datasets.

can be found in Fig S1-S4 in the Supplemental Material.

It is observed that using either PCA or Random Projection does not boost the performance of OCSVM. When 1,2,3, and 4 components (dimensions) were chosen, the accuracies of OCSVM on those transformed datasets are smaller than the accuracy of OCSVM on the original dataset. When the number of retained components (dimensions) were 5 and 6, the accuracies associated with PCA, normal RP, Bernoulli RP, and Achlioptas RP are equal to the original accuracy.

By contrast, using data transformation can improve the performance of LMDD, COPOD, and HBOS to varying degrees. Concerning COPOD, the COPOD + PCA case attains 25.3%, 29.2%, 25.5%, 26.3%, 23.8%, and 22% with 1, 2, 3, 4, 5, and 6 retained components respectively which are 2%−9% higher than the original result of COPOD. When using RP, COPOD can improve its performance in all cases except COPOD + Achlioptas with 1 dimension. The most improved case is by using Normal RP in which the accuracy of COPOD increases by up to 13.8%.

Significant improvements can be observed in the cases associated with HBOS. By using Bernoulli RP, the accuracy of HBOS can improve by up to more than 30% with 2, 3, 4, 5, and 6 dimensions. Normal RP also can boost the performance of HBOS + Normal with smaller increases from 10% to 30%. HBOS + Achlioptas meanwhile does not show any improvement when 1, 3, 4, and 5 dimensions were used. Finally, the results associated with LMDD method show a different pattern compared to those of OCSVM, COPOD, and HBOS. While using 1 component or dimension helped to significantly increase the accuracy of LMDD (more than 50% increase), the results are even poorer than the original one when more than 1 component or dimension was used.

To sum up, the experiments demonstrate that using a data transformation method like PCA or RP can improve the performance of an

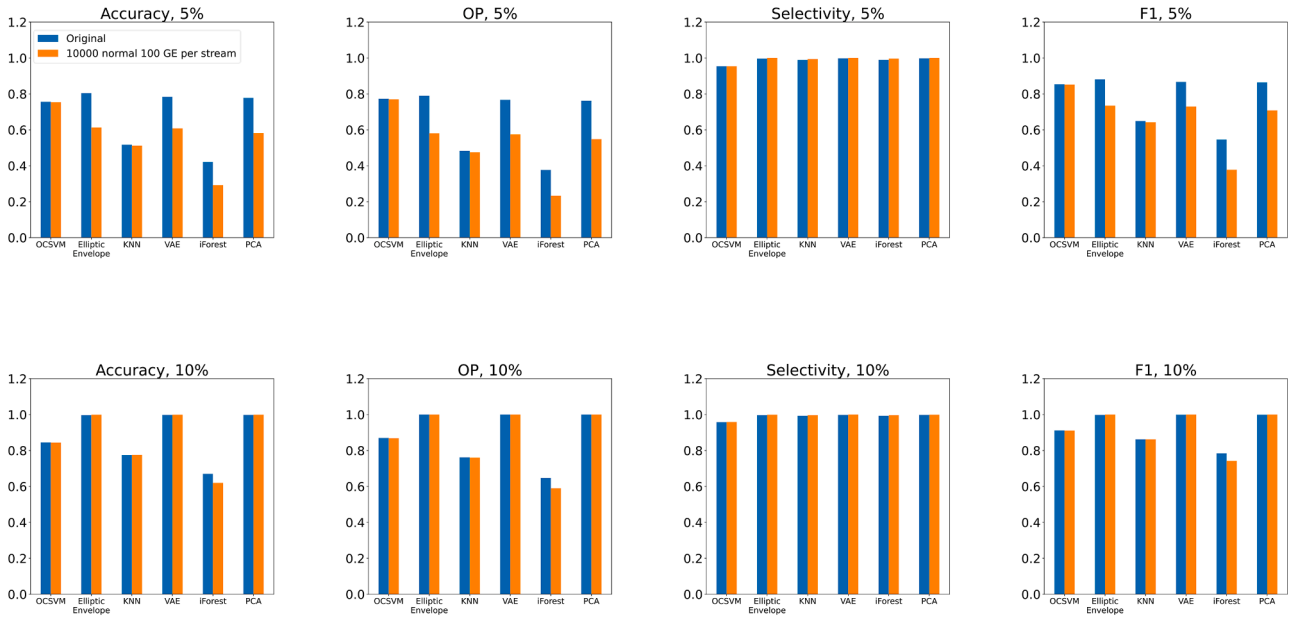


Fig. 9. The performance of 6 selected ADMs on P3 datasets.

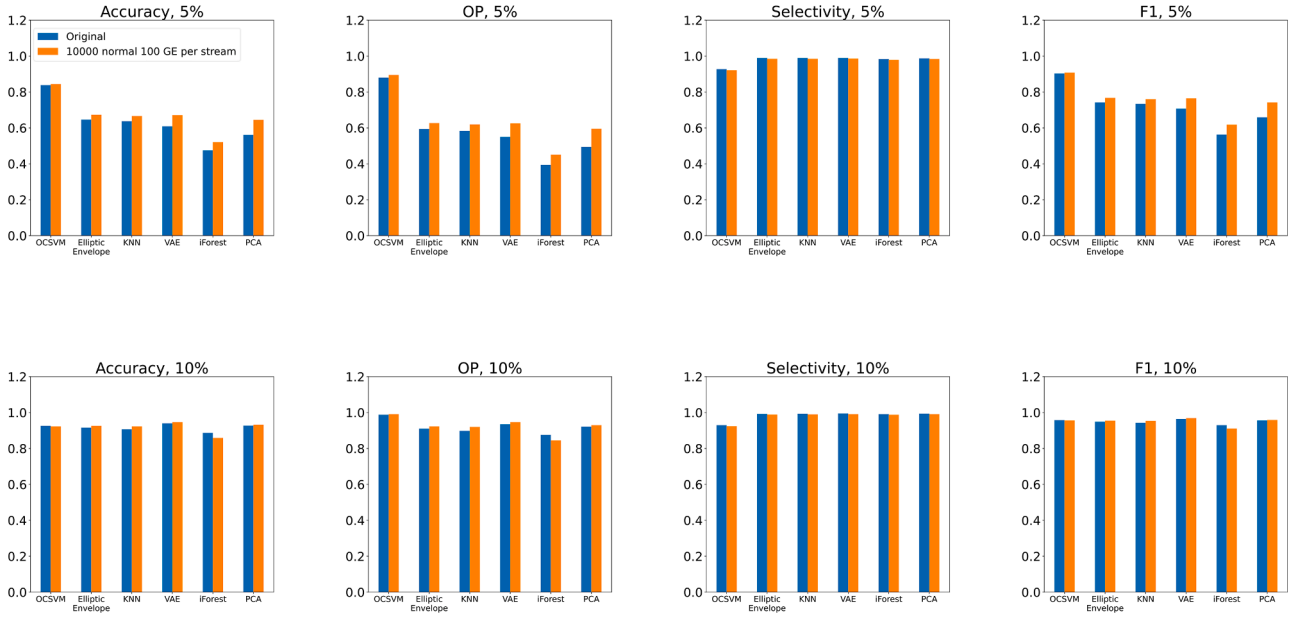


Fig. 10. The performance of 6 selected ADMs on P4 datasets.

ADM with different significances. For the best-performing methods data transformation/dimensional reduction does not significantly improve GED performance. For the poorest performing methods, it does help, however not significantly to challenge the best-performing methods. In practice, a data transformation method can be used on the measurement data before it is fed into an ADM. To determine which data transformation methods will be used for a particular dataset and ADM, we can evaluate them on a validation set and compare their performances. It is widely recognized that if a method performs well on the validation set, it can likely obtain high performance on the test data.

4.3. The impacts of size of training data

We investigated impacts of the size of training dataset on the performance of ADMs for the GED. Figs. 7-10 show the performances of the top 6 ADMs namely OCSVM, KNN, VAE, PCA, Elliptic Envelope, and iForest on 4 systems with two different training sizes: training dataset 1 (1000 non-GEs and m*10 observations where GEs are present on each stream) and training dataset 2 (10,000 non-GEs observations and m*100 observations where GEs are present on each stream). Dataset 2 is 10 times larger than dataset 1 however the ratio of GEs to non-GEs has been maintained.

It can be seen that using different numbers of training samples may impact the performance of some methods. On P1 and P2 system, the performance of VAE and OCSVM improve slightly when more training samples are available. In detail, on P1 system, the OP of VAE is 34.1% (5%) and 84.5% (10%) on the training dataset 2 which is better than the OP of VAE on the training dataset 1 by 1.5% (5%) and 3.5% (10%). On P2 dataset, the OP of OCSVM slightly increases from 82.98% (5%) and 96.57% (10%) (on the training dataset 1) to 83.8% (5%) and 97.11%

Table. 8
Training times in seconds of 19 ADMs on the 16 datasets with ±5% of GEs.

	iForest	LOF	OCSVM	Elliptic envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO-GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	1.305	0.018	0.509	1.124	0.018	0.007	0.007	1.984	2.086	0.083	21.547	22.866	65.76	20.133	0.002	14.338	3.205	1.62	15.822
P2	1.621	0.211	1.645	2.824	0.213	0.02	0.02	1.208	3.835	0.111	33.155	37.903	114.112	32.808	0.004	61.214	9.978	0.004	29.247
P3	2.291	0.653	7.814	6.959	0.632	0.068	0.068	2.518	43.699	0.166	59.59	66.647	212.882	57.741	0.015	348.248	36.219	0.009	98.283
P4	1.587	0.27	1.850	2.738	0.27	0.019	0.019	1.26	3.976	0.109	33.106	37.098	112.13	32.373	0.004	61.06	9.517	0.004	28.924
P5	1.829	0.565	3.382	2.669	0.549	0.032	0.032	1.845	10.353	0.125	41.099	46.923	144.518	41.073	0.008	119.046	16.208	0.005	47.783
P6	1.802	0.254	2.104	2.658	0.254	0.026	0.026	1.388	6.256	0.119	34.415	38.961	125.714	34.386	0.007	90.988	12.924	0.005	37.563
P7	2.106	0.794	3.906	2.642	0.793	0.048	0.048	2.378	22.266	0.143	53.854	59.924	173.333	48.913	0.011	225.782	24.667	0.007	71.152
P8	2.194	0.489	24.763	3.740	0.488	0.057	0.057	2.243	30.489	0.155	58.392	65.13	189.047	52.981	0.012	298.654	29.785	0.008	83.83
P9	2.364	0.514	35.307	3.124	0.513	0.067	0.067	2.366	40.829	0.162	57.789	66.526	205.137	57.462	0.015	348.882	36.187	0.009	98.305
P10	2.461	2.175	8.644	5.454	2.174	0.078	0.078	4.285	57.075	0.172	67.503	74.958	222.549	61.116	0.035	445.025	41.6	0.01	115.818
P11	2.325	0.214	25.595	3.722	0.214	0.068	0.067	2.094	43.022	0.16	58.086	66.114	208.727	57.186	0.014	346.958	35.685	0.009	97.639
P12	2.277	1.314	7.423	6.943	1.302	0.067	0.067	3.172	43.931	0.159	58.546	68.098	209.438	57.752	0.015	345.688	35.247	0.009	99.274
P13	2.956	0.485	11.483	10.314	0.218	0.119	0.119	2.69	106.615	0.194	75.707	85.827	271.187	74.181	0.024	760.507	66.486	0.014	166.521
P14	4.313	0.456	99.574	28.237	0.497	0.271	0.257	4.168	441.898	0.263	120.266	133.56	399.759	107.38	0.067	2580.398	182.662	0.026	359.666
P15	4.866	0.627	52.368	17.752	0.707	0.360	0.36	4.992	679.947	0.295	139.425	154.057	464.937	124.001	0.092	3938.922	267.139	0.033	481.33
P16	10.016	0.953	895.679	97.350	1.122	1.040	1.06	8.176	4176.322	0.317	193.118	209.082	667.99	155.374	0.248	25,283.23	1229.203	1.31	1629.394
Ave	2.895	0.625	73.878	12.391	0.623	0.147	0.147	2.923	357.037	0.171	69.100	77.105	236.701	63.429	0.036	2204.309	127.295	0.193	216.284

Table. 9
Testing times in seconds of 19 ADMs on the 16 datasets with ±5% of GEs.

	iForest	LOF	OCSVM	Elliptic envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO-GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.598	0.042	0.387	0.027	0.355	0.038	0.038	0.607	0.561	0.089	0.332	0.403	0.279	0.302	0.028	13.2	3.275	0.028	9.632
P2	0.858	0.236	1.181	0.05	0.854	0.087	0.086	1.236	0.32	0.158	0.4	0.43	0.401	0.39	0.051	61.181	10.254	0.051	29.214
P3	1.442	0.729	4.551	0.167	1.9	0.242	0.245	2.623	4.185	0.292	0.704	0.762	0.687	0.728	0.165	347.675	36.2	0.102	98.096
P4	0.836	0.295	1.215	0.047	0.91	0.085	0.083	1.293	0.299	0.148	0.396	0.429	0.388	0.393	0.048	60.832	9.682	0.048	28.728
P5	1.055	0.606	2.097	0.125	1.372	0.123	0.122	1.867	0.828	0.191	0.494	0.532	0.493	0.489	0.13	118.862	16.586	0.064	47.628
P6	1.075	0.429	1.565	0.119	1.114	0.104	0.103	1.55	0.52	0.177	0.528	0.631	0.462	0.488	0.118	90.932	12.922	0.055	37.496
P7	1.317	0.958	3.133	0.141	1.917	0.174	0.171	2.507	1.946	0.227	0.593	0.629	0.598	0.606	0.141	225.906	25.145	0.083	71.07
P8	1.374	0.574	8.056	0.16	1.613	0.204	0.199	2.299	2.928	0.254	0.735	0.823	0.653	0.689	0.149	298.822	29.848	0.091	83.628
P9	1.53	0.738	8.96	0.157	1.855	0.231	0.229	2.592	4.042	0.273	0.7	0.749	0.715	0.716	0.162	348.13	36.242	0.101	97.967
P10	1.587	2.519	5.357	0.168	3.81	0.265	0.261	4.567	5.629	0.287	0.746	0.787	0.741	0.746	0.165	444.382	42.18	0.109	115.108
P11	1.487	0.372	7.897	0.158	1.491	0.231	0.23	2.236	4.012	0.263	0.687	0.75	0.685	0.69	0.158	346.812	35.813	0.139	97.649
P12	1.426	1.526	4.548	0.174	2.677	0.233	0.23	3.411	4.03	0.27	0.717	0.745	0.703	0.688	0.166	347.482	36.341	0.139	98.878
P13	2.038	0.347	8.522	0.206	9.291	0.392	0.389	5.545	13.01	0.337	0.93	0.96	0.879	0.895	0.195	761.461	66.344	0.137	166.14
P14	3.162	0.687	31.042	0.303	27.897	0.831	0.832	10.606	68.014	0.487	1.351	1.401	1.314	1.33	0.28	2494.167	183.64	0.226	359.973
P15	3.544	0.917	30.346	0.345	37.21	1.223	1.2	30.006	128.758	0.56	1.538	1.633	1.561	1.566	0.345	3914.821	264.872	0.278	479.979
P16	7.539	1.443	198.911	0.639	288.202	2.807	2.793	140.906	1025.422	0.831	2.227	2.348	2.121	2.101	0.63	25,459.72	1161.52	0.546	1623.655
Ave	1.929	0.776	19.861	0.187	23.904	0.454	0.451	13.366	79.032	0.303	0.817	0.876	0.793	0.801	0.183	2208.399	123.179	0.137	215.303

Table. 10
The F1 Score of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±5% of GEs.

	iForest	OCSVM	Elliptic envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.522	0.780	0.507	0.520	0.481	0.460	0.237	0.237	0.237	0.237	0.066
P2	0.509	0.870	0.672	0.661	0.617	0.591	0.414	0.622	0.498	0.622	0.495
P3	0.546	0.854	0.881	0.649	0.867	0.864	0.503	0.656	0.591	0.656	0.945
P4	0.563	0.903	0.742	0.734	0.707	0.659	0.686	0.828	0.782	0.828	0.658
P5	0.368	0.832	0.454	0.465	0.410	0.433	0.847	0.898	0.879	0.898	0.383
P6	0.204	0.649	0.195	0.195	0.190	0.189	0.192	0.451	0.332	0.451	0.069
P7	0.299	0.800	0.459	0.489	0.387	0.401	0.349	0.690	0.596	0.690	0.422
P8	0.331	0.957	0.309	0.287	0.304	0.267	0.185	0.487	0.414	0.487	0.257
P9	0.265	0.960	0.328	0.199	0.282	0.267	0.291	0.641	0.563	0.641	0.350
P10	0.323	0.804	0.463	0.456	0.289	0.260	0.552	0.838	0.775	0.838	0.711
P11	0.241	0.960	0.212	0.272	0.212	0.218	0.213	0.657	0.567	0.657	0.314
P12	0.342	0.875	0.754	0.696	0.718	0.729	0.676	0.854	0.818	0.854	0.837
P13	0.238	0.723	0.214	0.234	0.199	0.210	0.095	0.592	0.442	0.592	0.194
P14	0.465	0.973	0.368	0.416	0.091	0.086	0.522	0.896	0.845	0.896	0.872
P15	0.367	0.814	0.095	0.240	0.073	0.071	0.195	0.795	0.665	0.795	0.489
P16	0.512	0.990	0.027	0.111	0.000	0.000	0.717	0.971	0.945	0.971	0.990
Ave	0.381	0.859	0.418	0.414	0.364	0.357	0.417	0.694	0.622	0.694	0.503

Table. 11
The F1 Score of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±10% of GEs.

	iForest	OCSVM	Elliptic envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.906	0.906	0.901	0.906	0.877	0.870	0.573	0.573	0.573	0.573	0.598
P2	0.880	0.944	0.929	0.924	0.936	0.937	0.758	0.829	0.784	0.829	0.893
P3	0.784	0.912	0.998	0.862	0.999	0.999	0.694	0.778	0.739	0.778	0.998
P4	0.930	0.958	0.949	0.943	0.964	0.956	0.922	0.955	0.953	0.955	0.936
P5	0.738	0.937	0.829	0.789	0.842	0.847	0.913	0.932	0.923	0.932	0.824
P6	0.260	0.705	0.280	0.262	0.283	0.263	0.194	0.491	0.370	0.491	0.113
P7	0.470	0.856	0.693	0.693	0.689	0.686	0.619	0.809	0.751	0.809	0.681
P8	0.653	0.957	0.756	0.571	0.746	0.716	0.460	0.703	0.638	0.703	0.789
P9	0.461	0.960	0.646	0.476	0.652	0.637	0.549	0.768	0.721	0.768	0.673
P10	0.588	0.928	0.856	0.799	0.870	0.861	0.822	0.946	0.931	0.946	0.932
P11	0.488	0.960	0.686	0.482	0.678	0.691	0.551	0.860	0.804	0.860	0.825
P12	0.588	0.934	0.955	0.856	0.977	0.976	0.857	0.932	0.920	0.932	0.977
P13	0.367	0.786	0.384	0.386	0.375	0.381	0.176	0.708	0.550	0.708	0.421
P14	0.633	0.975	0.858	0.765	0.856	0.853	0.835	0.955	0.918	0.955	0.961
P15	0.516	0.895	0.512	0.557	0.490	0.486	0.470	0.883	0.799	0.883	0.930
P16	0.682	0.990	0.039	0.418	0.000	0.000	0.894	0.977	0.964	0.977	0.995
Ave	0.622	0.913	0.704	0.668	0.702	0.697	0.643	0.819	0.771	0.819	0.784

Table. 12
The Selectivity of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±5% of GEs.

	iForest	OCSVM	Elliptic envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.919	0.826	0.926	0.916	0.917	0.913	0.901	0.901	0.901	0.901	0.972
P2	0.972	0.915	0.982	0.977	0.981	0.979	0.980	0.958	0.960	0.958	0.997
P3	0.990	0.954	0.997	0.989	0.998	0.998	0.990	0.971	0.976	0.971	0.997
P4	0.983	0.927	0.989	0.989	0.989	0.987	0.984	0.949	0.959	0.949	0.997
P5	0.972	0.925	0.981	0.978	0.979	0.977	0.913	0.897	0.901	0.897	0.991
P6	0.892	0.882	0.894	0.896	0.885	0.889	0.883	0.879	0.878	0.879	0.894
P7	0.949	0.936	0.974	0.978	0.967	0.969	0.971	0.942	0.946	0.942	0.988
P8	0.962	0.917	0.967	0.960	0.968	0.968	0.954	0.942	0.944	0.942	0.978
P9	0.966	0.923	0.983	0.950	0.979	0.978	0.951	0.940	0.939	0.940	0.982
P10	0.985	0.951	0.998	0.992	0.995	0.995	0.980	0.955	0.956	0.955	0.991
P11	0.967	0.923	0.974	0.956	0.978	0.977	0.965	0.943	0.947	0.943	0.976
P12	0.984	0.954	0.998	0.995	0.996	0.996	0.992	0.965	0.972	0.965	0.995
P13	0.957	0.945	0.958	0.956	0.959	0.957	0.956	0.948	0.947	0.948	0.959
P14	0.987	0.962	1.000	0.995	1.000	1.000	0.996	0.975	0.975	0.975	0.994
P15	0.981	0.971	0.990	0.985	0.993	0.991	0.983	0.970	0.972	0.970	0.985
P16	0.993	0.980	1.000	1.000	0.000	0.000	0.998	0.983	0.986	0.983	0.995
Ave	0.966	0.931	0.976	0.970	0.912	0.911	0.962	0.945	0.947	0.945	0.981

(10%) (on the training dataset 2). A similar pattern can be found in the performance of VAE and OCSVM relating to accuracy and F1 score. On P4 system, the performance of 5 methods (VAE, KNN, PCA, OCSVM, and Elliptic Envelope) improve with more training samples except for iForest on the P4 10% GE dataset.

By contrast, on P1 and P2 system, Elliptic Envelope, KNN, PCA, and iForest performed slightly poorer on training dataset 2 than on training set 1. For example, the F1 score of Elliptic Envelope is 50.69% (5%) and 90.14% (10%) on the training dataset 1 and 50.1% (5%) and 89.9% (10%) on the training dataset 2 of P1 system, respectively. The F1 score of iForest decreased from 52.22% (5%) and 90.56% (10%) to 50.5% (5%) and 89.6% (10%) when more training samples were used. On P3 system with 5% GE, the reductions in the performance Elliptic Envelope, KNN, PCA, and iForest are more significant than those on the P1 and P2

Table. 13
The Selectivity of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±10% of GEs.

	iForest	OCSVM	Elliptic envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.960	0.852	0.961	0.960	0.956	0.956	0.966	0.966	0.966	0.966	0.998
P2	0.987	0.924	0.987	0.987	0.989	0.989	0.990	0.972	0.977	0.972	0.999
P3	0.993	0.958	0.997	0.993	0.998	0.998	0.992	0.979	0.984	0.979	0.997
P4	0.991	0.930	0.992	0.993	0.995	0.994	0.991	0.962	0.971	0.962	1.0
P5	0.987	0.939	0.991	0.989	0.991	0.992	0.925	0.907	0.912	0.907	0.997
P6	0.914	0.892	0.926	0.917	0.921	0.921	0.908	0.891	0.898	0.891	0.940
P7	0.974	0.941	0.987	0.986	0.988	0.989	0.988	0.951	0.960	0.951	0.993
P8	0.982	0.917	0.990	0.984	0.990	0.990	0.983	0.965	0.969	0.965	0.995
P9	0.976	0.923	0.989	0.978	0.988	0.989	0.972	0.952	0.953	0.952	0.992
P10	0.991	0.963	0.999	0.996	0.999	0.999	0.987	0.961	0.966	0.961	0.994
P11	0.986	0.923	0.994	0.981	0.995	0.995	0.992	0.963	0.968	0.963	0.995
P12	0.991	0.959	0.998	0.996	0.998	0.997	0.994	0.968	0.976	0.968	0.995
P13	0.973	0.953	0.980	0.976	0.982	0.981	0.980	0.956	0.959	0.956	0.985
P14	0.992	0.963	1.000	0.998	1.000	1.000	0.998	0.978	0.979	0.978	0.995
P15	0.987	0.975	0.998	0.995	0.999	0.999	0.995	0.975	0.979	0.975	0.995
P16	0.995	0.980	1.000	1.000	0.000	0.000	0.999	0.984	0.986	0.984	0.994
Ave	0.980	0.937	0.987	0.983	0.924	0.924	0.979	0.958	0.963	0.958	0.9915

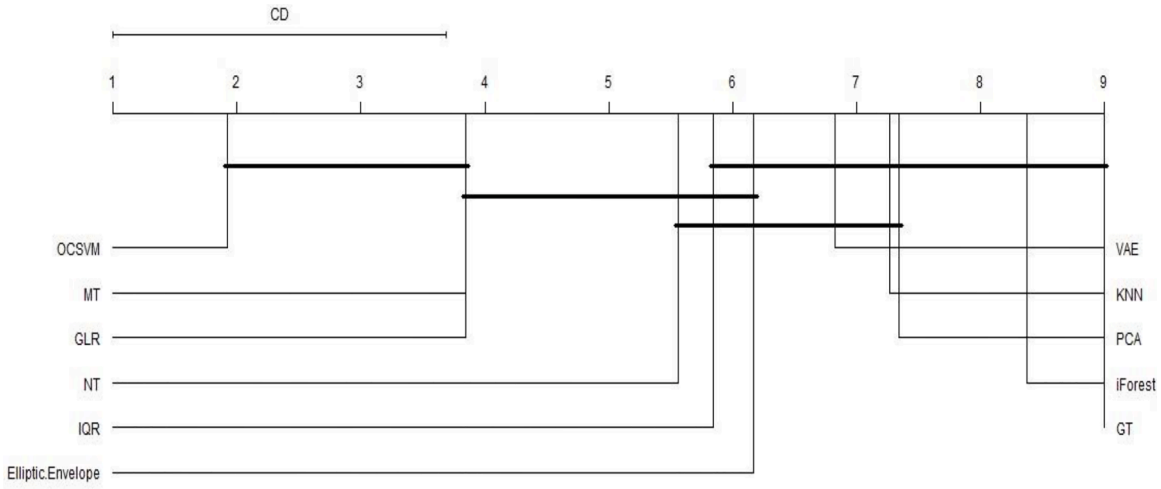


Fig.11. Nemenyi test results on Accuracy when comparing 6 ADMs to 5 statistical tests
OCSVM > NT, IQR, Elliptic Envelope, VAE, KNN, PCA, iForest, GT
MT, GLR> VAE, KNN, PCA, iForest, GT
NT > iForest, GT.

system. VAE for example obtained 76.72% of OP on the training dataset 1 while this method only obtained 57.5% of OP on the training dataset 2.

To sum up, the size of training datasets may affect the performances of ADMs on experimental systems with different significances. Some methods obtained better results while some methods performed poorer when more training samples were used. When applying an ADM on a training dataset, the method tries to distinguish between GE and non-GE samples to form a detection-making strategy on test samples. Using more training samples may change the discriminative ability of the training data that affect the performance of the detection-making strategy of ADMs.

4.4. The training and testing time

Table 8 shows the training time of 19 ADMs on the 16 datasets with ±5% of GEs. The experiments were conducted on a PC with a Core i7 processor and 128GB RAM. For iForest, LOF, OCSVM, Elliptic Envelope, KNN, ECOD, COPOD, ABOD, LODA, Autoencoder, VAE, Deep SVDD, PCA, COF, HBOS, in most cases their training times never exceed five minutes, except for OCSVM and COF on the P16 dataset, which took 896 and 1230 s (or 15 and 20 min) for training respectively.

Meanwhile, ROD, SO-GAAL, LMDD, and SOD took much higher training time than the others, requiring a training time of 357.037, 236.701, 2204.309, and 216.284 s on average, respectively. The training time of ROD is the smallest among these 4 methods on the P1-P15 datasets. In detail, for the P1 dataset, ROD took around 2 s for training while SO-GAAL, LMDD, and SOD required 65.76, 14.338, and 15.822 s respectively. From P2 to P15, SO-GAAL required at most 464.937 s (on P15) which is lower than the required training time of ROD on P15 by more than 200 s, while SOD had the highest training time at around 360 s on the P14 dataset. For the P16 dataset, SO-GAAL took just around 10 min for training, while ROD and SOD required 4176.322 and 1629.394 s, which is higher by 6.15 and 3.39 times compared to the time for training on the P15 dataset. Finally, the sets of training time of LMDD on the P1-P13 dataset were from 14 s to almost 13 min, but on P14, P15, and P16 dataset the required times for training were more than 25,000 s or almost 7 h, which is the highest among all methods on all datasets.

Table 9 shows the testing time for the 19 ADMs on the 16 datasets. The methods which did not require more than 5 min for training also had small testing time. Even though SO-GAAL required from 1 to 7.73 min for training, it only required less than 1 s for testing on all datasets. For the ROD method, on P1 to P15, the required testing time was from more than 0.5 s to around 2 min. However, on the P16 dataset, the testing time of this method was up to 17 min. The testing time required for LMDD and SOD were comparable to the training time, especially for LMDD on

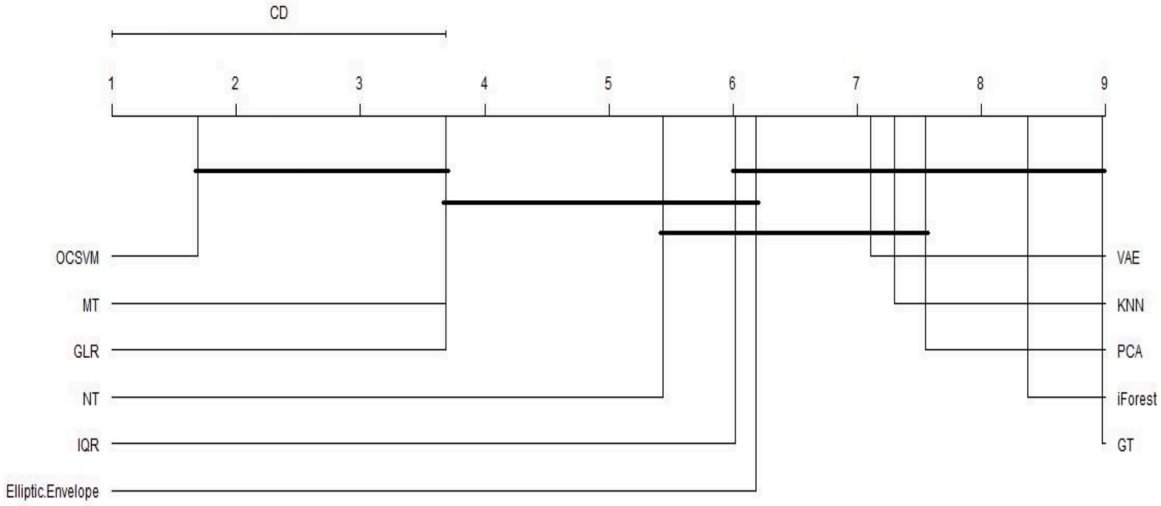


Fig. 12. Nemenyi test results on OP when comparing 6 ADMs to 5 statistical tests
OCSVM > NT, IQR, Elliptic Envelope, VAE, KNN, PCA, iForest, GT
MT, GLR> VAE, KNN, PCA, iForest, GT
NT > iForest, GT.

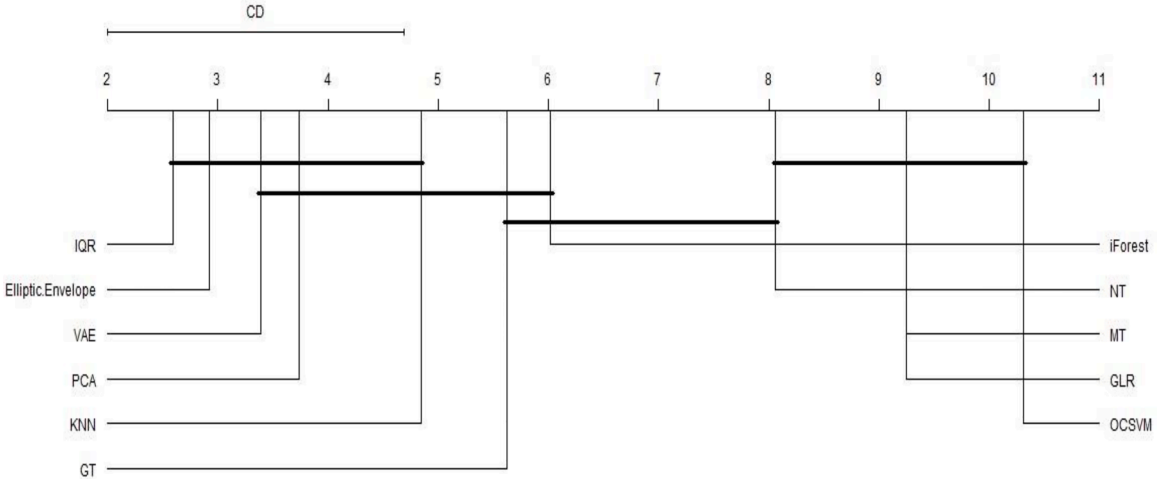


Fig. 13. Nemenyi test results on Selectivity when comparing 6 ADMs to 5 statistical tests
IQR, Elliptic Envelope > GT, iForest, NT, MT, GLR, OCSVM
VAE, KNN, PCA > NT, MT, GLR, OCSVM
GT, iForest > MT, GLR, OCSVM.

the P16 dataset which required roughly 7 h.

The reported training and testing time of ADMs on the 16 datasets demonstrate that these methods are practical when used to detect GEs.

4.5. Comparison with conventional methods

We chose the top 6 ADMs in the above experiments to compare with several conventional GED methods. Five statistical tests namely GT, MT, NT, IQR, and GLR were selected as the baselines. The statistics of MT GT, NT, and GLR were calculated based on uncertainties of streams provided in the original studies (see Fig S5-S20). The significant level of these tests was set to 0.05. We did not compare with other serial elimination, serial compensation, and collective compensation methods because those methods aim to investigate the location of GEs on the measurement data while ADMs output the binary results indicating whether GE presents on the measurement data or not. We did not compare with other supervised ML methods such as the ensemble methods in Nguyen et al. (2020), Dobos et al. (2021) or probabilistic-based methods such as Bayesian Network in Yuan et al. (2015) because those methods require ground truth information of GE for the training data which is not necessary to train unsupervised ADMs. Tables 10-13 show the F1 Score and Selectivity of the top 6 ADMs and 5 statistical tests on the experimental datasets. The Accuracy and OP of these methods can be found in Table.S.5-Table.S.8 in the Supplemental Material.

The Friedman test shows that there are differences in the performances of the top 6 ADMs and the 5 statistical tests on all four performance metrics. Figs. 11-14 show the Nemenyi post-hoc test when comparing each pairwise of experimental methods. It can be seen that OCSVM ranks first, followed by MT, GLR, NT, IQR, and Elliptic Envelop while PCA, iForest, and GT are the 3 poorest methods on Accuracy, F1 score, and OP.

For F1 Score, Accuracy, and OP, OCSVM is better than NT, IQR, Elliptic Envelope, VAE, KNN, PCA, iForest, and GT. MT and GLR perform exactly similar and are better than VAE, KNN, PCA, iForest, and GT. While NT is better than iForest and GT and there are no statistical differences in the performances of MT, GLR, NT, IQR, and Elliptic Envelop.

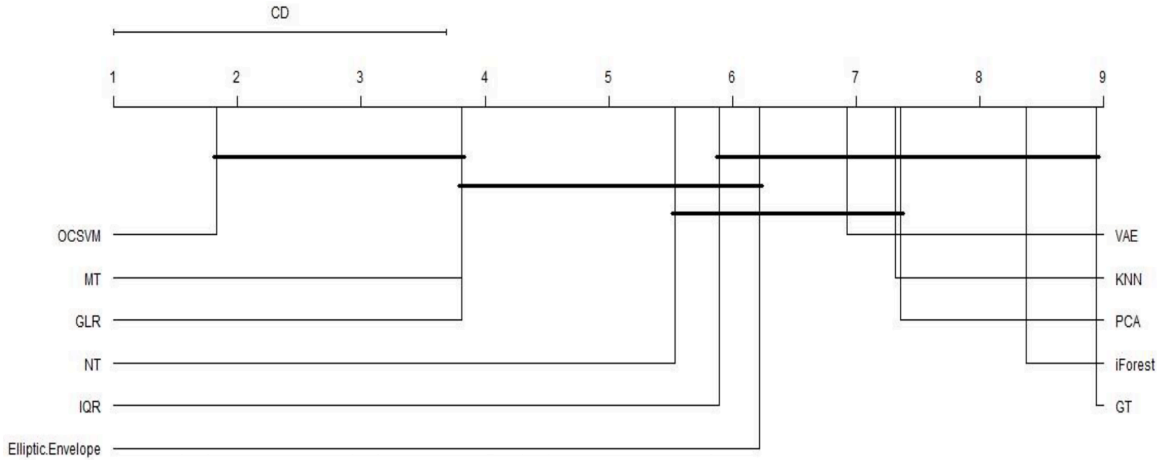


Fig. 14. Nemenyi test results on F1 Score when comparing 6 ADMs to 5 statistical tests
OCSVM > NT, IQR, Elliptic Envelope, VAE, KNN, PCA, iForest, GT
MT, GLR> VAE, KNN, PCA, iForest, GT
NT > iForest, GT.

Table A1
Information of experimental datasets (with ±5% of GEs or ±10% of GEs) generated from 16 systems.

	# of training instances	# of testing instances	# of dimensions
P1	4000	4000	3
P2	7000	7000	6
P3	13,000	13,000	12
P4	7000	7000	6
P5	9000	9000	8
P6	8000	8000	7
P7	11,000	11,000	10
P8	12,000	12,000	11
P9	13,000	13,000	12
P10	14,000	14,000	13
P11	13,000	13,000	12
P12	13,000	13,000	12
P13	17,000	17,000	16
P14	25,000	25,000	24
P15	29,000	29,000	28
P16	51,000	51,000	50

On average, OCSVM obtained 85.9% and 91.3% for F1 Score on 5% and 10% GE dataset respectively. Those values are about 15% and 10% better than the average values of the second-best methods (MT and GLR). On 5% GE datasets, OCSVM performed poorer than MT and GLR on P5 and P10 system only. By contrast, iForest, GT, KNN, and PCA obtained 38.1%, 41.7%, 41.4%, and 35.7% of F1 Score on average. Although the performances of those methods improved on 10% GE datasets, they are still the poorest among all 10 experimental methods.

For Selectivity, IQR and Elliptic Envelope are better than GT, iForest, NT, MT, GLR, and OCSVM based on the Nemenyi test. Meanwhile, VAE, KNN, and PCA are better than NT, MT, GLR, and OCSVM; GT and iForest are better than MT, GLR, and OCSVM. OCSVM predicted the presence of GE on non-GE samples a little more than some methods, resulting in its slightly low value of Selectivity (Precision). The difference between the Selectivity of OCSVM compared to those of the best ADMs (IQR) is about 5% only. IQR is the confident method in which its Selectivity is 98.1% and 99.15% on average on 5% and 10% GE datasets. Elliptic Envelop performed well when considering the Selectivity as this method obtained 97.6% and 98.7% on average on 5% and 10% GE datasets. The NT, MT, and GLR performed poorer than all ADMs except OCSVM. That means those statistical tests have a strong tendency to label as positive (i.e., has GE) a sample that is negative (i.e., has no GE).

To conclude, OCSVM, MT, and GLR are the top three methods in the experiments concerning OP, F1 Score, and Accuracy. However, these methods label a sample as positive a little more than the other methods like IQR and Elliptic Envelop, resulting in slightly lower values of Selectivity (Precision).

The ADMs can be effectively used to detect GEs on the measurement data since two ADMs namely OCSVM and Elliptic Envelop outperform the others as well as the statistical tests. It is noted that ADMs can be used if historical measurement data is available, making ADMs more practical than supervised ML methods for the GED task (Reddy and Mavrovouniotis, 1998) (Nguyen et al., 2020) since labelling the training samples to train supervised ML (i.e., associating each measurement data with GE information) requires huge cost and effort. It should also be noted that the statistical tests require uncertainty estimation of each stream in the system based on experts' input and are therefore disadvantageous compared to ADMs.

Although OCSVM outperforms the others on F1 Score, Accuracy, and OP, this method outputted a higher value of false positive than the others, which makes this method less reliable because of labelling non-GE samples as GE samples. This also happened with MT and GLR's outputs.

5. Conclusions

In this paper, we have introduced an application of ADMs to detect GEs on measurement data when historical data is available. We first reviewed the developments of GE techniques including statistical tests, serial elimination, serial compensation, collective compensation methods, and several ML-based methods. We also conducted an intensive review of ML-based and DL-based ADMs. The experimental framework was introduced with several steps: data preparation, training data generation, training a GED model, and detecting on testing data.

The experiments were conducted with 19 selected ADMs on synthetic datasets generated from 16 systems in the literature. We generated 16 training datasets and 32 testing datasets with 5% and 10% of GE on each stream in each system. We used 4 performance metrics namely Accuracy, Overall Power (OP), Selectivity, and F1 Score to report the performance of the 19 ADMs on the testing datasets.

The experimental results indicate that:

- The top 6 ADMs including OCSVM, Elliptic Envelop, KNN, PCA, VAE, and iForest were obtained from the experimental results and Nemenyi test results. OCSVM outperformed the other ADMs based on OP, Accuracy, and F1 Score. OCSVM achieved slightly lower Selectivity

Table A2
Abbreviations used in the paper.

Abbreviation	Meaning	Abbreviation	Meaning
GE	Gross Error	SOD	Subspace Outlier Degree
GED	Gross Error Detection	OCGAN	One-class Generative Adversarial Network
ML	Machine Learning	LODA	Lightweight online detector of anomalies
ADM	Anomaly Detection Method	HBOS	Histogram-based Outlier Score
DL	Deep Learning	KNN	K-Nearest Neighbours
AI	Artificial Intelligence	SO-GAAL	Single-Objective Generative Adversarial Active Learning
MGE	Multiple Gross Error	Deep SVDD	Deep Support Vector Data Description
GT	Global Test	DR	Data Reconciliation
NT	Constraint Test or Nodal Test	GAN	Generative Adversarial Networks
MT	Measurement Test	IQR	Interquartile Range
GLR	Generalized Likelihood Ratio	RNN	Recurrent Neural Network
PCA	Principal Component Analysis	DBN	Deep Belief Network
PCO	Particle Swarm Optimization	CNN	Convolutional Neural Network
GMM	Gaussian Mixture Model	ALAD	Adversarially Learned Anomaly Detection
OCSVM	One-class Support Vector Machine	CSI	Contrasting Shifted instances
SVDD	Support Vector Data Description	iForest	Isolation Forest
PCA	Principal Component Analysis	OP	Overall power
VAE	Variational Autoencoder	TP	True Positive
LOF	Local Outlier Factor	FP	False Positive
ABOD	Angle-Based Outlier Detection	FN	False Negative
ROD	Rotation-based Outlier Detection	TN	True Negative
COF	Connectivity-based Outlier Factor	df	Degrees of freedom
LMDD	Linear Method for Deviation Detection	Ave	Average
ECOD	Empirical-Cumulative-distribution-based Outlier Detection	CD	Critical Difference
COPOD	Copula-based Outlier Detector	RP	Random Projection

than several ADMs however this was not sufficient to undermine its overall performance.

- We observed the sensitivity of KNN and PCA based on their performances in the experiments. Their performance should be evaluated on a validation set before deciding whether they are selected for applications. Besides, methods like ECOD and COPOD performed poorly because they are ineffective in detecting dependency anomalies. These methods thus should not be used to detect the GEs for systems including dependent relationships between stream variances.
- The ADMs can potentially be applied to detect GEs on the measurement data when historical data is available. Based on the experimental results, OCSVM should be the first choice for GED applications, especially for linear systems.
- We compared the top 6 ADMs to the 5 statistical tests namely IQR, NT, MT, GLR, and GT. Experimental results showed that OCSVM, MT, and GLR are the top 3 methods based on OP, Accuracy, and F1 Score. Like OCSVM, MT, and GLR obtained slightly high values of false positive that make their detection results slightly less reliable. The statistical tests need the information of balance equations and information uncertainty in the calculation, which is usually based on a subjective opinion from experts and is therefore not applicable to automated AD. However, when the historical data is not available to train ADMs, MT or GLR should be the first choice for applications.
- Applying data transformation to the measurement data before training with an ADM can increase the performance of some ADM, especially the worst-performance ones. In this study, we applied two data transformation and dimension reduction methods namely PCA and Random Projection to the measurement data. Experimental results showed that the performances of some ADMs improved with different significances. In practice, the data transformation method and its hyper-parameters can be determined by evaluating the performance of ADM on a validation set.
- The size of the training data affects the performance of the ADMs to varying degrees. In some methods like OCSVM or VAE, using more training samples to train an ADM could improve its performance. On the other hand, the performances of some methods like iForest or PCA are downgraded when more training samples were used.

Some future work can be considered (i) searching for the optimal hyper-parameters of an ADM on a particular dataset to further improve its performance (ii) exploring approaches to reduce the high false positive of some ADMs to make their detection results more reliable (iii) developing ADMs-based ensemble to obtain better results than using a single ADM.

CRediT authorship contribution statement

Daniel Dobos: Methodology, Data curation, Writing – original draft. **Tien Thanh Nguyen:** Conceptualization, Methodology, Writing – original draft, Supervision. **Truong Dang:** Validation, Visualization. **Allan Wilson:** Methodology, Supervision, Writing – review & editing. **Helen Corbett:** Validation, Writing – review & editing. **John McCall:** Methodology, Supervision, Project administration. **Phil Stockton:** Conceptualization, Project administration, Writing – review & editing.

Declaration of Competing Interest

The authors declares that they have no conflict of interest.

Data availability

Data will be made available on request.

Acknowledgement

This work was supported by Innovate UK [grant number KTP011488].

Appendix

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.compchemeng.2023.108263.

References

Abdallah, A., Maarof, M.A., Zainal, A., 2016. Fraud detection system: a survey. J. Netw. Comput. Appl. 68, 90–113.
Aggarwal, C.C., 2017. Proximity-based outlier detection. Outlier Analysis, pp. 111–147.
Akcay, S., Atapour-Abarghouei, A., Breckon, T.P., 2018. GANomaly: semi supervised anomaly detection via adversarial training. In: Asian Conference on Computer Vision, pp. 622–637.

Almardeny, Y., Boujnah, N., Grant, F., 2020. A novel outlier detection method for multivariate data. IEEE Trans. Knowl. Data Eng. 34, 4052–4062.
Amruthnath, N., Gupta, T., 2018. A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance. In: International Conference on Industrial Engineering and Applications, pp. 355–361.
Angiulli, F., Pizzuti, C., 2002. Fast outlier detection in high dimensional spaces. In: European Conference on Principles of Data Mining and Knowledge Discovery, pp. 15–27.
Arning, A., Agrawal, R., Raghavan, P., 1996. A linear method for deviation detection in large databases. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 164–169.
Aytekin, C., Ni, X., Cricri, F., Aksu, E., 2018. Clustering and unsupervised anomaly detection with l2 normalized deep auto-encoder representations. In: International Joint Conference on Neural Networks (IJCNN), pp. 1–6.
Bakar, Z.A., Mohemad, R., Ahmad, A., Deris, M.M., 2006. A comparative study for outlier detection techniques in data mining. In: IEEE Conference on Cybernetics and Intelligent Systems, pp. 1–6.
Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J., 2000. LOF: identifying density-based local outliers. In: ACM SIGMOD International Conference on Management of Data, pp. 93–104.
E. Candes, X. Li, Y. Ma, J. Wright. Robust principal component analysis? Recovering low-rank matrices from sparse errors, IEEE Sensor Array and Multichannel Signal Processing Workshop. (2010) 201–204.
Carrera, D., Boracchi, G., Foi, A., 2015. Detecting anomalous structures by convolutional sparse models. In: International Joint Conference on Neural Networks (IJCNN), pp. 1–8.
Cerri, O., Nguyen, T.Q., Pierini, M., Spiropulu, M., Vlimant, J.R., 2019. Variational autoencoders for new physics mining at the large hadron collider. J. High Energy Phys. 36.
Chalapathy, R., Menon, A.K., Chawla, S., 2017. Robust, deep and inductive anomaly detection. Mach. Learn. Knowl. Disc. Databases 36–51.
Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: a survey. ACM Comput. Surv. 41.
Congli, M., Hongye, S., Jian, C., 2006. An NT-MT combined method for gross error detection and data reconciliation. Chin. J. Chem. Eng. 14, 592–596.
Crowe, C.M., Garcia Campos, Y.A., 1983. Reconciliation of process flow rates by matrix projection part I: linear case. Am. Inst. Chem. Eng. J. 29, 881–888.
Crowe, C.M., 1989. Test of maximum power for detection of gross errors in process constraints. Am. Inst. Chem. Eng. J. 35, 869–872.
do Valle, E.C., Kalid, R.A., Secchi, A.R., Kiperstok, A., 2018. Collection of benchmark test problems for data reconciliation and gross error detection and identification. Comput. Chem. Eng. 111, 134–148.
Dobos, D., Nguyen, T.T., McCall, J., Wilson, A., Stockton, P., Corbett, H., 2021. Weighted ensemble of gross error detection methods based on particle swarm optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 307–308.
Erfani, S.M., Rajasegarar, S., Karunasekera, S., Leckie, C., 2016. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. Pattern Recognit. 58, 121–134.
Fujimaki, R., Yairi, T., Machida, K., 2005. An approach to spacecraft anomaly detection problem using kernel feature space. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 401–410.
Garcia, S., Herrera, F., 2008. An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. J. Mach. Learn. Res. 9, 2579–2596.
Gerber, E.F., Auret, L., Aldrich, C., 2014. The application of classification methods to the gross error detection problems. In: Proceedings of the 19th World Congress The international federation of Automatic Control.
Golan, I., El-Yaniv, R., 2018. Deep anomaly detection using geometric transformations. Adv. Neural Inf. Process. Syst. 31.
Goldstein, M., Dengel, A., 2012. Histogram-based outlier score (HBOS): a fast unsupervised anomaly detection algorithm. In: 35th German Conference on Artificial Intelligence (KI), pp. 59–63.
Gong, D., Liu, L., Le, V., Saha, B., Mansour, M., Venkatesh, S., et al., 2019. Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsupervised anomaly detection. In: International Conference on Computer Vision (ICCV), pp. 1705–1714.
Haakon, R., Augustin, S., Jennifer, R., Christophe, D., 2007. Sensitivity of PCA for traffic anomaly detection. SIGMETRICS Perform. Eval. Rev. 35 (1), 109–120. June.
Hardin, J., Rocke, D., 2004. Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. Comput. Stat. Data Anal. 44, 625–638.
Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., 2012. Deep neural networks for acoustic modelling in speech recognition. Signal Process. Magazine.
H. Hojjati, T.K.K. Ho, N. Armanfard, Self-supervised anomaly detection: a survey and outlook, arXiv:2205.05173 [cs]. (2022).
Huang, F.J., LeCun, Y., 2006. Large-scale learning with SVM and convolutional for generic object categorization. In: Conference on Computer Vision and Pattern Recognition, pp. 284–291.
Jiang, X., Liu, P., Li, Z., 2014. Data reconciliation and gross error detection for operational data in power plants. Energy 75, 14–23.
Johnson, W., Lindenstrauss, J., 1984. Extensions of Lipschitz mapping into Hilbert space. In: Conference in Modern Analysis and Probability, Contemporary Mathematics, pp. 189–206.
Jordache, C., Narashimhan, S., 1999. Data Reconciliation and Gross Error Detection. An Intelligent Use of Process Data. Gulf Professional Publishing.
Keller, J.-Y., Darouach, M., Krzakala, G., 1994. Fault detection of multiple biases or process leaks in linear steady state systems. Comput. Chem. Eng. 18, 1001–1004.
Kim, J., Scott, C.D., 2012. Robust kernel density estimation. J. Mach. Learn. Res. 13, 2529–2565.
D.P. Kingma, M. Welling, An introduction to variational autoencoders, arXiv.1906.02691 [cs]. (2019).
Kriegel, H.P., Schubert, M., Zimek, A., 2008. Angle-based outlier detection in high-dimensional data. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 444–452.
Kriegel, H.P., Kroger, P., Schubert, E., Zimek, A., 2009. Outlier detection in axis-parallel subspaces of high dimensional data. Lecture Notes in Computer Science, pp. 831–838.
Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 84–90.
Laurikkala, J., Juhola, M., Kentala, E., 2000. Informal identification of outliers in medical data. In: Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology, pp. 20–24.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444.
Li, Z., Zhao, Y., Botta, N., Ionescu, C., Hu, X., 2020. Copod: copula-based outlier detection. In: IEEE International Conference on Data Mining (ICDM), pp. 1118–1123.
Li, C.L., Sohn, K., Yoon, J., Pfster, T., 2021. Cutpaste: self-supervised learning for anomaly detection and localization. In: Computer Vision and Pattern Recognition (CVPR), pp. 9659–9669.
Li, Z., Zhao, Y., Hu, X., Botta, N., Ionescu, C., Chen, G., 2022. Ecod: unsupervised outlier detection using empirical cumulative distribution functions. In: IEEE Transactions on Knowledge and Data Engineering.
Liu, F.T., Ting, K., Zhou, Z.H., 2009. Isolation forest. In: IEEE International Conference on Data Mining, pp. 413–422.
Liu, Y., Li, Z., Zhou, C., Jiang, Y., Sun, J., Wang, M., et al., 2020. Generative adversarial active learning for unsupervised outlier detection. IEEE Trans. Knowl. Data Eng. 32, 1517–1528.
Liu, B., Wang, D., Lin, K., Tan, P.N., Zhou, J., 2021. RCA: a deep collaborative autoencoder approach for anomaly detection. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 1505–1511.
Loyola-Fuentes, J., Smith, R., 2019. Data reconciliation and gross error detection in crude oil pre-heat trains undergoing shell-side and tube-side fouling deposition. Energy 183, 368–384.
Lu, W., Cheng, Y., Xiao, C., Chang, S., Huang, S., Liang, B., et al., 2017. Unsupervised sequential outlier detection with deep architectures. IEEE Trans. Image Process. 26, 4321–4330.
Mah, R.S.H., Tamhane, A.C., 1982. Detection of gross errors in process data. Am. Inst. Chem. Eng. J. 28, 828–830.
Mah, R.S.H., Stanley, G., Downing, D., 1976. Reconciliation and rectification of process flow and inventory data. Ind. Eng. Chem. Process Des. Dev. 15, 175–183.
Narasimhan, S., Mah, R., 1987. Generalized likelihood ratio method for gross error identification. Am. Inst. Chem. Eng. J. 33, 1514–1521.
Nguyen, T.T., Dang, M.T., Liew, A.W.C., Bezdek, J.C., 2019. A weighted multiple classifier framework based on random projection. Inf. Sci 490, 36–58.
Nguyen, T.T., McCall, J., Wilson, A., Ocheil, L., Corbett, H., Stockton, P., 2020. Evolved ensemble of detectors for gross error detection. In: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, pp. 281–282.
Nogita, S., 1972. Statistical test and adjustment of process data. IEC Proc. Des. Dev.
P. Oza, V.M. Patel, One-class convolutional neural network, arXiv:1901.08688v1 [cs]. (2019).
Patcha, A., Park, J.M., 2007. An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput. Netw. 51, 3448–3470.
Pedregosa, F., 2011. Scikit-learn: machine Learning in Python. J. Mach. Learn. Res. (JMLR) 12, 2825–2830.
Perera, P., Nallapati, R., Xiang, B., 2019. OCGAN: one-class novelty detection using GANs with constrained latent representations. In: Computer Vision and Pattern Recognition, pp. 2893–2901.
Pevny, T., 2016. LODA: lightweight on-line detector of anomalies. Mach. Learn. 102, 275–304.
Pidhorskyi, S., Almohsen, R., Adjeroh, D.A., Doretto, G., 2018. Generative probabilistic novelty detection with adversarial autoencoders. In: International Conference on Neural Information Processing Systems, pp. 6823–6834.
Prata, D.M., Schwaab, M., Lima, E.L., Pinto, J.C., 2010. Simultaneous robust data reconciliation and gross error detection through particle swarm optimisation for an industrial polypropylene reactor. Chem. Eng. Sci. 65, 4943–4954.
Principi, E., Vesperini, F., Squartini, S., Piazza, F., 2017. Acoustic novelty detection with adversarial autoencoders. In: International Joint Conference on Neural Networks (IJCNN).
Reddy, V.N., Mavrovouniotis, M.L., 1998. An input-training Neural Network approach for gross error detection and sensor replacement. Chem. Eng. Res. Des. 76, 478–489.
Reilly, P., Carpani, R., 1963. Application of statistical theory of adjustments to material balances. In: Proceedings of the 13th Canadian chemical engineering conference.
Ripps, D.L., 1965. Adjustment of Experimental Data. Chem. Eng. Prog. Symp.
Rollins, D., Davis, J., 1992. Unbiased estimation of gross errors in process measurements. Am. Inst. Chem. Eng. J. 38, 563–572.
Romagnoli, J.A., Stephanopolous, G., 1981. Rectification of process measurement data in the presence of gross errors. Chem. Eng. Sci.
Rosenberg, J., Mah, R.S.H., Iordache, C., 1987. Evaluation of schemes for detecting and identifying gross errors in process data. Ind. Eng. Chem. Res. 26, 555–564.
Rousseeuw, P., Driessen, K., 1999. A fast algorithm for the minimum covariance determinant estimator. Technometrics 41, 212–223.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., et al., 2018. Deep one-class classification. In: International Conference on Machine Learning, pp. 4393–4402.
Ruff, L., Kauffmann, J.R., Vandermeulen, R.A., Montavon, G., Samek, W., Kloft, M., et al., 2021. A unifying review of deep and shallow anomaly detection. Proc. IEEE 109.
Sánchez, M., Romagnoli, J., Jiang, Q., 1999. Simultaneous estimation of biases and leaks in process plants. Comput. Chem. Eng. 23, 841–857.
Sabokrou, M., Khalooei, M., Fathy, M., Adeli, E., 2018. Adversarially learned one-class classifier for novelty detection. In: Computer Vision and Pattern Recognition, pp. 3379–3388.
Schlegl, T., Seebock, P., Waldstein, S.M., Langs, G., Schmidt-Erfurth, U., 2019. f-AnoGAN: fast unsupervised anomaly detection with generative adversarial networks. Med. Image Anal. 54.
Shen, L., Li, Z., Kwok, J., 2020. Timeseries anomaly detection using temporal hierarchical one-class network. Adv. Neural Inf. Process. Syst. 33, 13016–13026.
Shyu, M.L., Chen, S.C., Sarinnapakorn, K., Chang, L., 2003. A novel anomaly detection scheme based on principal component classifier. In: Proceedings of International Conference on Data Mining.
Songqiao, H., Xiyang, H., Hailiang, H., Mingqi, J., Yue, Z., 2022. ADBench: anomaly detection benchmark. In: Neural Information Processing Systems (NeurIPS).
Tack, J., Mo, S., Jeong, J., Shin, J., 2020. CSI: novelty detection via contrastive learning on distributionally shifted instances. In: International Conference on Neural Information Processing Systems, pp. 11839–11852.
Tamhane, A.C., Iordache, C., Mah, R.S.H., 1988. A Bayesian approach to gross error detection in chemical process data- Part I: model Development. Chemometr. Intell. Lab. Syst. 4, 33–45.
Tang, J., Chen, Z., Fu, A.W.C., Cheung, D.W., 2002. Enhancing effectiveness of outlier detections for low density patterns. Advances in Knowledge Discovery and Data Mining, pp. 535–548.
Tax, D.M., Duin, R.P., 2004. Support vector data description. Mach. Learn. 54, 45–66.
Tong, H., Crowe, C.M., 1995. Detection of gross errors in data reconciliation by Principal component analysis. Am. Inst. Chem. Eng. J. 41, 1712–1722.
Wang, F., Jia, X.-p., Zheng, S.-q., Yue, J.-c., 2004. An improved MT-NT method for gross error detection and data reconciliation. Comput. Chem. Eng. 28, 2189–2192.
Wang, D., Yeung, D., Tsang, E., 2007. Structured one-class classification. IEEE Trans. Syst. Man Cybern. 36, 1283–1295.
Weiss, K., Khoshgoftaar, T.M., Wang, D., 2016. A survey of transfer learning. J. Big Data 3.
J. Weston, S. Chopra, A. Bordes, Memory networks, abs/1410.3916 [cs]. (2015).
Wu, Y., Lin, Y., Zhou, Z., Bolton, D.C., Liu, J., Johnson, P., 2019. Deep detect: a cascaded region-based densely connected network for seismic event detection. IEEE Trans. Geosci. Remote Sens. 57, 62–75.
Wu, P., Liu, J., Shen, F., 2020. A deep one-class neural network for anomalous event detection in complex scenes. IEEE Trans. Neural Netw. Learn. Syst. 31, 2609–2622.
Yang, Y., Ten, R., Jao, L., 1995. A study of gross error detection and data reconciliation in process industries. Comput. Chem. Eng. 19, 217–222.
Yuan, Y., Khatibisepehr, S., Huang, B., Li, Z., 2015. Bayesian method for simultaneous gross error detection and data reconciliation. Am. Inst. Chem. Eng. J. 61, 3232–3248.
Zenati, H., Romain, M., Foo, C.S., Lecouat, B., Chandrasekhar, V.R., 2018. Adversarially learned anomaly detection. In: IEEE International Conference on Data Mining (ICDM), pp. 727–736.
Zhang, X., Mu, J., Liu, H., Zong, L., Li, Y., 2022. Deep anomaly detection with self-supervised learning and adversarial training. Pattern Recognit. 121.
Zhao, Y., Nasrullah, Z., Li, Z., 2019. PyOD: a python toolbox for scalable outlier detection. J. Mach. Learn. Res. (JMLR) 20, 1–7.
Zhou, C., Paffenroth, R.C., 2017. Anomaly detection with robust deep autoencoders. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 665–674.

Supplemental Material

Paper: **A comparative study of anomaly detection methods for gross error detection problems**

Table S.1. The Selectivity of the19 experimental ADMs on the 16 datasets with ±5% of GEs

	iForest	LOF	OCSVM	Elliptic Envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO_GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.919	0.899	0.826	0.926	0.916	0.857	0.962	0.914	0.866	0.891	0.906	0.917	0.877	0.732	0.913	0.833	0.818	0.917	0.845
P2	0.972	0.974	0.915	0.982	0.977	0.944	0.992	0.978	0.979	0.947	0.978	0.981	0.924	0.741	0.979	0.933	0.955	0.947	0.976
P3	0.990	0.993	0.954	0.997	0.989	0.978	0.996	0.993	0.989	0.987	0.998	0.998	0.000	0.907	0.998	0.998	0.999	0.988	1.000
P4	0.983	0.985	0.927	0.989	0.989	0.942	0.998	0.988	0.988	0.982	0.987	0.989	0.000	0.700	0.987	0.977	0.912	0.969	0.969
P5	0.972	0.975	0.925	0.981	0.978	0.950	0.986	0.975	0.962	0.918	0.977	0.979	0.000	0.832	0.977	0.932	1.000	0.952	1.000
P6	0.892	0.890	0.882	0.894	0.896	0.884	0.897	0.888	0.878	0.895	0.886	0.885	0.889	0.870	0.889	0.884	0.936	0.889	1.000
P7	0.949	0.979	0.936	0.974	0.978	0.946	0.967	0.979	0.959	0.926	0.969	0.967	0.000	0.881	0.969	0.963	0.983	0.946	1.000
P8	0.962	0.946	0.917	0.967	0.960	0.953	0.977	0.954	0.961	0.954	0.967	0.968	0.000	0.900	0.968	0.937	0.996	0.945	1.000
P9	0.966	0.957	0.923	0.983	0.950	0.961	0.972	0.961	0.959	0.953	0.978	0.979	0.000	0.897	0.978	0.932	1.000	0.950	1.000
P10	0.985	0.995	0.951	0.998	0.992	0.978	0.991	0.992	0.983	0.949	0.994	0.995	0.953	0.848	0.995	0.957	1.000	0.976	1.000
P11	0.967	0.946	0.923	0.974	0.956	0.964	0.976	0.956	0.958	0.956	0.974	0.978	0.000	0.881	0.977	0.949	0.996	0.961	1.000
P12	0.984	0.995	0.954	0.998	0.995	0.973	0.995	0.993	0.984	0.952	0.996	0.996	0.000	0.951	0.996	0.986	0.996	0.974	1.000
P13	0.957	0.942	0.945	0.958	0.956	0.954	0.966	0.954	0.953	0.956	0.956	0.959	0.949	0.935	0.957	0.952	1.000	0.953	1.000
P14	0.987	0.957	0.962	1.000	0.995	0.984	0.994	0.995	0.987	0.982	1.000	1.000	0.000	0.935	1.000	0.973	1.000	0.981	1.000
P15	0.981	0.973	0.971	0.990	0.985	0.982	0.990	0.987	0.986	0.981	0.992	0.993	0.000	0.957	0.991	0.979	1.000	0.979	1.000
P16	0.993	0.992	0.980	1.000	1.000	0.992	0.995	0.993	0.992	0.984	0.000	0.000	0.000	0.968	0.000	0.988	1.000	0.990	1.000
Average	0.966	0.962	0.931	0.976	0.970	0.953	0.978	0.969	0.962	0.951	0.910	0.912	0.287	0.871	0.911	0.948	0.974	0.957	0.987

Table S.2. The Selectivity of the 19 experimental ADMs on the 16 datasets with ±10% of GEs

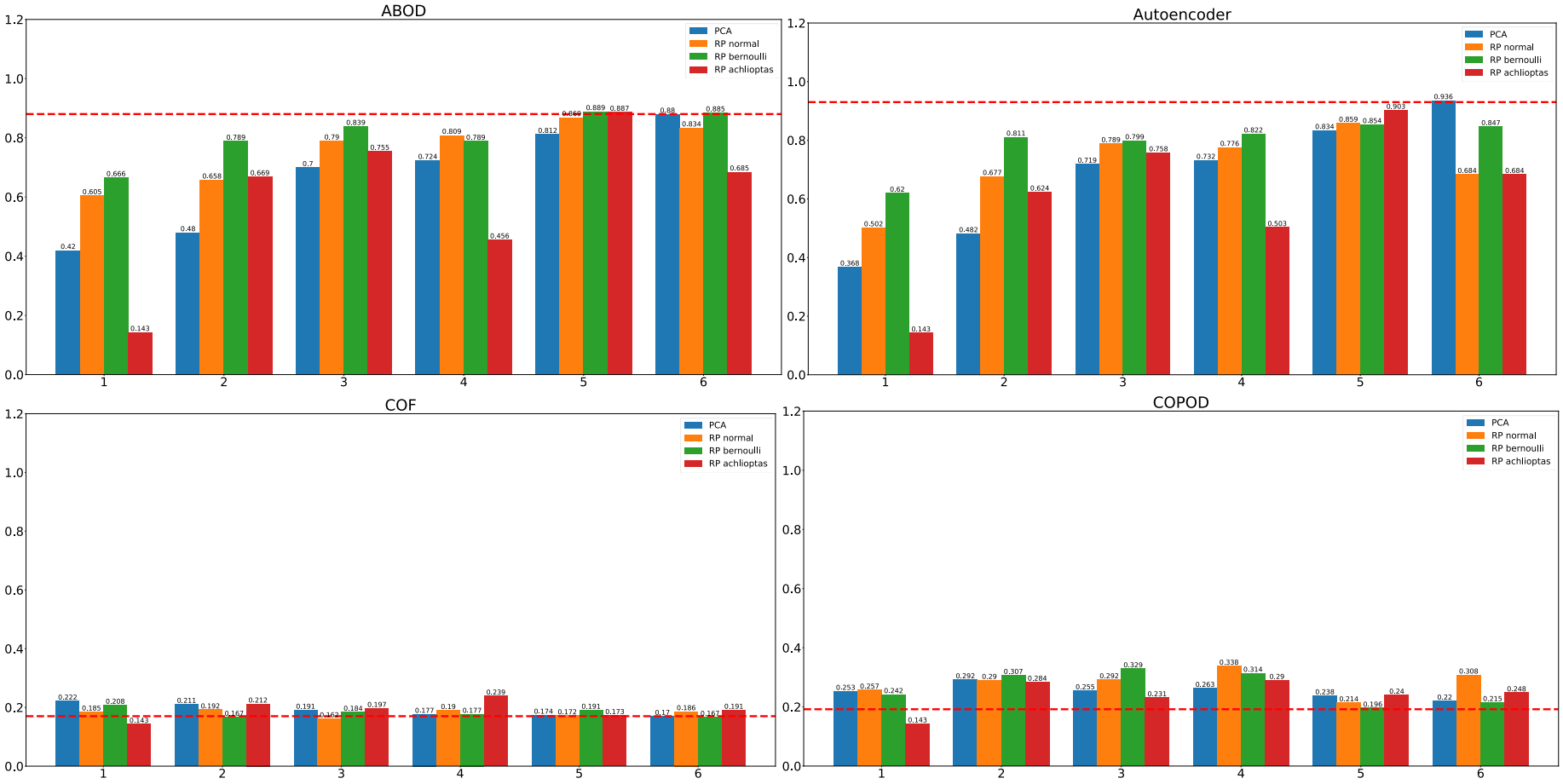
	iForest	LOF	OCSVM	Elliptic Envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO_GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.960	0.959	0.852	0.961	0.960	0.872	0.997	0.965	0.993	0.948	0.960	0.956	0.910	0.930	0.956	0.890	0.869	0.959	0.792
P2	0.987	0.985	0.924	0.987	0.987	0.945	0.997	0.987	0.997	0.926	0.989	0.989	0.931	0.960	0.989	0.975	0.887	0.968	0.895
P3	0.993	0.996	0.958	0.997	0.993	0.971	0.996	0.995	0.995	0.980	0.997	0.998	0.957	0.971	0.998	0.995	0.994	0.992	1.000
P4	0.991	0.990	0.930	0.992	0.993	0.946	0.994	0.993	0.997	0.992	0.994	0.995	0.000	0.865	0.994	0.997	0.878	0.976	0.889
P5	0.987	0.987	0.939	0.991	0.989	0.963	1.000	0.987	0.994	0.977	0.992	0.991	0.000	0.912	0.992	0.975	0.974	0.967	0.989
P6	0.914	0.911	0.892	0.926	0.917	0.907	0.942	0.926	0.907	0.918	0.916	0.921	0.912	0.884	0.921	0.899	0.994	0.915	1.000
P7	0.974	0.989	0.941	0.987	0.986	0.965	0.993	0.987	0.981	0.967	0.988	0.988	0.934	0.927	0.989	0.991	0.979	0.956	1.000
P8	0.982	0.980	0.917	0.990	0.984	0.968	0.996	0.986	0.981	0.979	0.990	0.990	0.000	0.908	0.990	0.969	0.985	0.960	1.000
P9	0.976	0.979	0.923	0.989	0.978	0.964	0.983	0.981	0.975	0.957	0.989	0.988	0.000	0.928	0.989	0.947	0.999	0.963	1.000
P10	0.991	0.998	0.963	0.999	0.996	0.982	0.996	0.994	0.987	0.992	0.999	0.999	0.940	0.907	0.999	0.992	0.991	0.986	0.992
P11	0.986	0.976	0.923	0.994	0.981	0.973	0.999	0.979	0.988	0.979	0.995	0.995	0.000	0.934	0.995	0.983	0.998	0.976	1.000
P12	0.991	0.997	0.959	0.998	0.996	0.977	0.997	0.994	0.992	0.976	0.998	0.998	0.974	0.986	0.997	1.000	0.996	0.979	1.000
P13	0.973	0.923	0.953	0.980	0.976	0.977	0.985	0.976	0.974	0.979	0.980	0.982	0.953	0.924	0.981	0.969	1.000	0.971	1.000
P14	0.992	0.961	0.963	1.000	0.998	0.986	0.995	0.999	0.990	0.995	1.000	1.000	0.000	0.977	1.000	0.994	1.000	0.984	1.000
P15	0.987	0.970	0.975	0.998	0.995	0.988	0.995	0.996	0.990	0.991	0.999	0.999	0.000	0.962	0.999	0.985	1.000	0.986	1.000
P16	0.995	0.993	0.980	1.000	1.000	0.990	0.994	0.994	0.991	0.983	0.000	0.000	0.000	0.977	0.000	0.989	1.000	0.992	1.000
Average	0.980	0.975	0.937	0.987	0.983	0.961	0.991	0.984	0.983	0.971	0.924	0.924	0.469	0.935	0.924	0.972	0.972	0.971	0.972

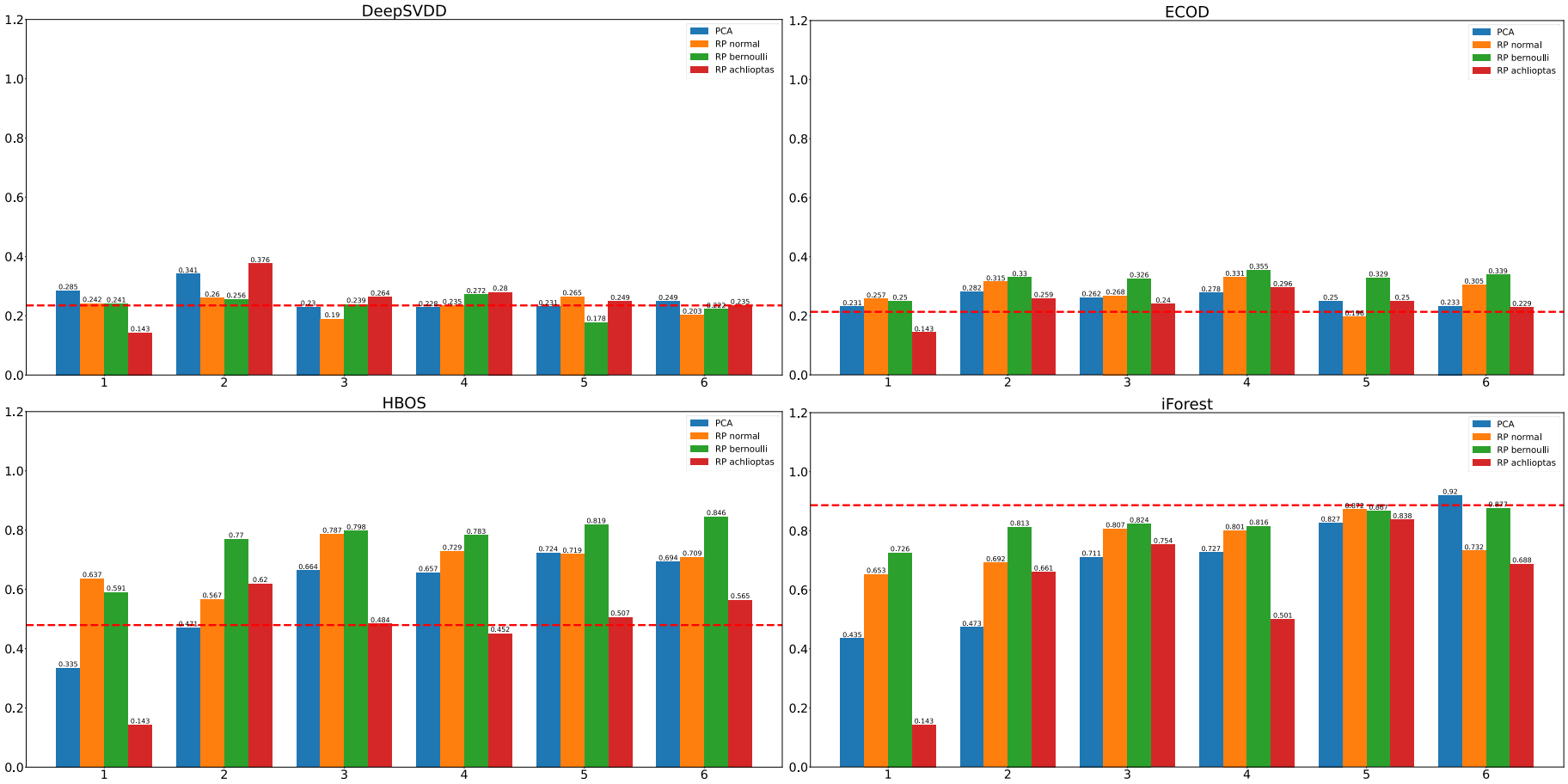
Table S.3. The F1 Score of the 19 experimental ADMs on the 16 datasets with ±5% of GEs

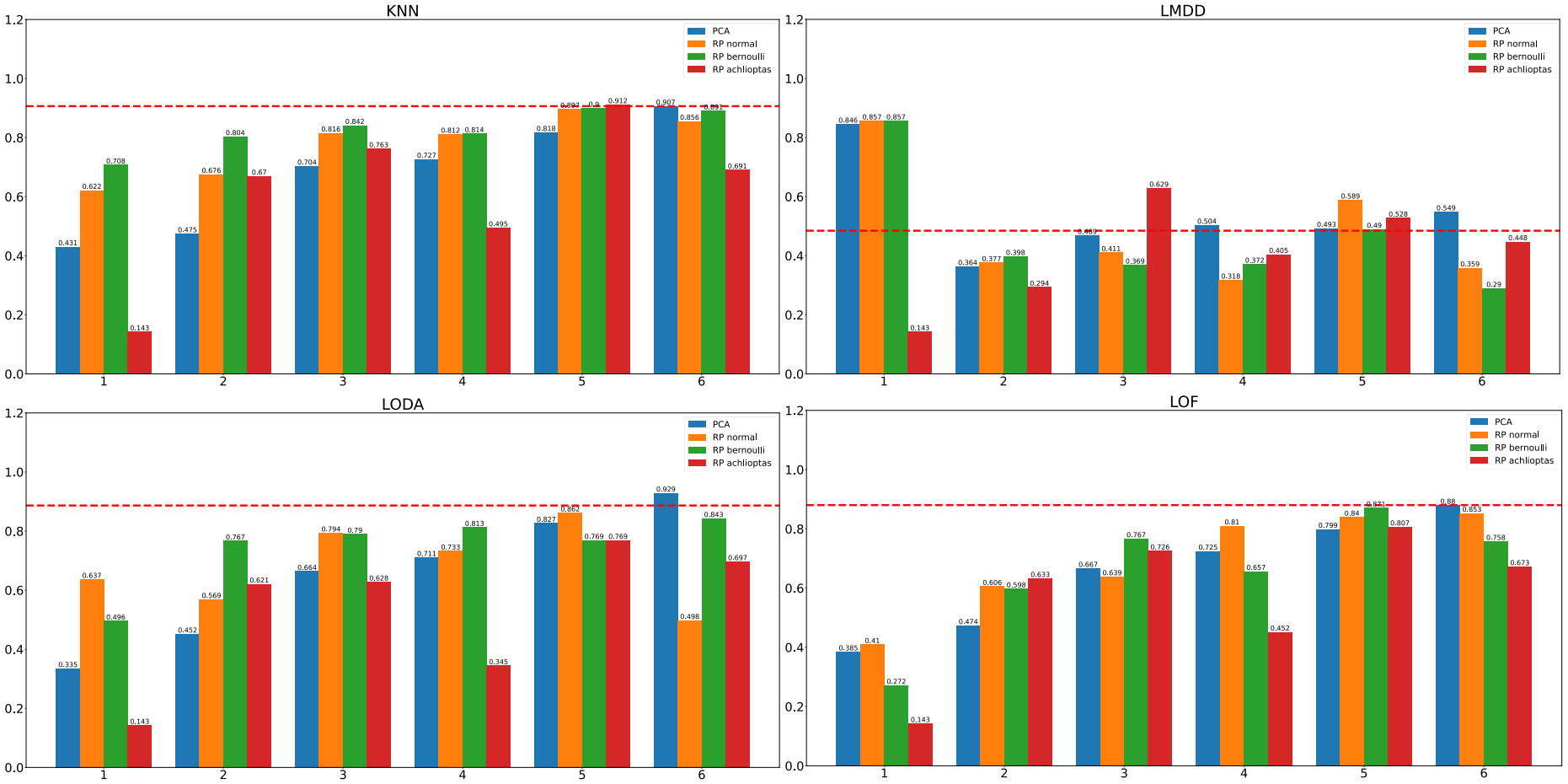
	iForest	LOF	OCSVM	Elliptic Envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO_GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.522	0.458	0.780	0.507	0.520	0.205	0.264	0.468	0.174	0.351	0.454	0.481	0.381	0.149	0.460	0.478	0.119	0.465	0.106
P2	0.509	0.625	0.870	0.672	0.661	0.183	0.157	0.633	0.154	0.420	0.581	0.617	0.315	0.068	0.591	0.469	0.068	0.466	0.089
P3	0.546	0.682	0.854	0.881	0.649	0.173	0.089	0.635	0.167	0.312	0.841	0.867	0.000	0.165	0.864	0.158	0.315	0.529	0.075
P4	0.563	0.729	0.903	0.742	0.734	0.164	0.148	0.718	0.171	0.565	0.685	0.707	0.000	0.065	0.659	0.476	0.073	0.463	0.098
P5	0.368	0.439	0.832	0.454	0.465	0.211	0.196	0.428	0.178	0.175	0.370	0.410	0.000	0.091	0.433	0.597	0.045	0.367	0.060
P6	0.204	0.187	0.649	0.195	0.195	0.190	0.192	0.201	0.176	0.185	0.189	0.190	0.164	0.189	0.189	0.522	0.084	0.202	0.001
P7	0.299	0.460	0.800	0.459	0.489	0.206	0.201	0.432	0.217	0.168	0.370	0.387	0.000	0.173	0.401	0.422	0.359	0.282	0.074
P8	0.331	0.255	0.957	0.309	0.287	0.212	0.194	0.237	0.166	0.225	0.264	0.304	0.000	0.183	0.267	0.607	0.372	0.269	0.076
P9	0.265	0.238	0.960	0.328	0.199	0.174	0.193	0.230	0.183	0.193	0.254	0.282	0.000	0.173	0.267	0.374	0.352	0.284	0.226
P10	0.323	0.469	0.804	0.463	0.456	0.162	0.107	0.448	0.134	0.156	0.247	0.289	0.247	0.071	0.260	0.489	0.059	0.385	0.060
P11	0.241	0.200	0.960	0.212	0.272	0.195	0.179	0.243	0.157	0.220	0.186	0.212	0.000	0.134	0.218	0.317	0.322	0.261	0.153
P12	0.342	0.700	0.875	0.754	0.696	0.192	0.161	0.730	0.164	0.205	0.654	0.718	0.000	0.292	0.729	0.416	0.079	0.386	0.070
P13	0.238	0.074	0.723	0.214	0.234	0.188	0.185	0.174	0.180	0.222	0.199	0.199	0.199	0.283	0.210	0.385	0.042	0.243	0.034
P14	0.465	0.054	0.973	0.368	0.416	0.211	0.152	0.128	0.206	0.210	0.080	0.091	0.000	0.194	0.086	0.549	0.263	0.314	0.259
P15	0.367	0.038	0.814	0.095	0.240	0.229	0.199	0.058	0.125	0.213	0.068	0.073	0.000	0.320	0.071	0.238	0.286	0.242	0.250
P16	0.512	0.025	0.990	0.027	0.111	0.261	0.238	0.023	0.219	0.212	0.000	0.000	0.000	0.236	0.000	0.488	0.136	0.337	0.026
Average	0.381	0.352	0.859	0.418	0.414	0.197	0.178	0.362	0.173	0.252	0.340	0.364	0.082	0.174	0.357	0.437	0.186	0.343	0.104

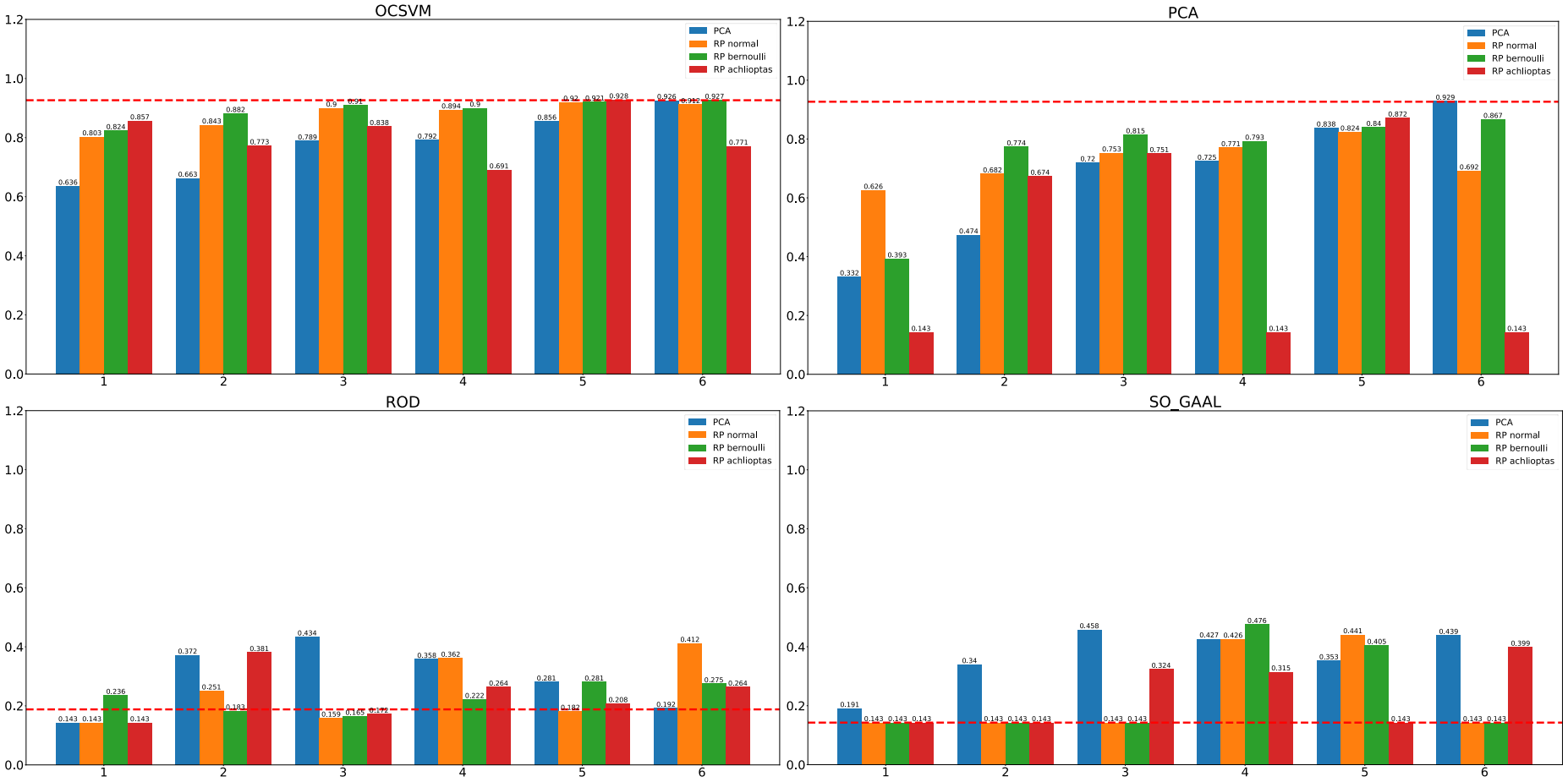
Table S.4. The F1 Score of the 19 experimental ADMs on the 16 datasets with ±10% of GEs

	iForest	LOF	OCSVM	Elliptic Envelope	KNN	ECOD	COPOD	ABOD	ROD	LODA	Autoencoder	VAE	SO_GAAL	Deep SVDD	PCA	LMDD	COF	HBOS	SOD
P1	0.906	0.844	0.906	0.901	0.906	0.202	0.223	0.890	0.175	0.708	0.856	0.877	0.533	0.567	0.870	0.496	0.139	0.860	0.182
P2	0.880	0.903	0.944	0.929	0.924	0.177	0.123	0.902	0.114	0.328	0.929	0.936	0.316	0.433	0.937	0.566	0.094	0.641	0.163
P3	0.784	0.864	0.912	0.998	0.862	0.179	0.084	0.835	0.189	0.380	0.999	0.999	0.183	0.593	0.999	0.410	0.152	0.636	0.068
P4	0.930	0.925	0.958	0.949	0.943	0.160	0.109	0.925	0.101	0.930	0.958	0.964	0.000	0.222	0.956	0.571	0.071	0.570	0.124
P5	0.738	0.684	0.937	0.829	0.789	0.183	0.153	0.724	0.157	0.492	0.819	0.842	0.000	0.229	0.847	0.558	0.081	0.514	0.129
P6	0.260	0.237	0.705	0.280	0.262	0.198	0.193	0.275	0.190	0.280	0.265	0.283	0.174	0.190	0.263	0.470	0.130	0.249	0.041
P7	0.470	0.673	0.856	0.693	0.693	0.179	0.158	0.657	0.217	0.403	0.683	0.689	0.266	0.261	0.686	0.466	0.196	0.343	0.093
P8	0.653	0.516	0.957	0.756	0.571	0.182	0.137	0.512	0.142	0.579	0.706	0.746	0.000	0.215	0.716	0.450	0.336	0.382	0.136
P9	0.461	0.509	0.960	0.646	0.476	0.175	0.152	0.496	0.193	0.347	0.630	0.652	0.000	0.224	0.637	0.206	0.351	0.407	0.158
P10	0.588	0.838	0.928	0.856	0.799	0.170	0.098	0.723	0.173	0.636	0.856	0.870	0.208	0.160	0.861	0.393	0.066	0.604	0.076
P11	0.488	0.399	0.960	0.686	0.482	0.151	0.108	0.467	0.141	0.375	0.643	0.678	0.000	0.245	0.691	0.270	0.131	0.417	0.092
P12	0.588	0.880	0.934	0.955	0.856	0.176	0.138	0.871	0.164	0.485	0.971	0.977	0.423	0.763	0.976	0.474	0.040	0.480	0.105
P13	0.367	0.080	0.786	0.384	0.386	0.177	0.138	0.287	0.169	0.362	0.372	0.375	0.267	0.263	0.381	0.209	0.139	0.337	0.149
P14	0.633	0.057	0.975	0.858	0.765	0.205	0.135	0.430	0.181	0.685	0.852	0.856	0.000	0.474	0.853	0.379	0.074	0.382	0.071
P15	0.516	0.043	0.895	0.512	0.557	0.217	0.163	0.226	0.145	0.355	0.476	0.490	0.000	0.386	0.486	0.416	0.247	0.409	0.321
P16	0.682	0.034	0.990	0.039	0.418	0.271	0.243	0.075	0.225	0.202	0.000	0.000	0.000	0.347	0.000	0.567	0.059	0.456	0.035
Average	0.622	0.530	0.913	0.704	0.668	0.188	0.147	0.581	0.167	0.472	0.688	0.702	0.148	0.348	0.697	0.431	0.144	0.480	0.121


ABOD
PCA
RP normal
RP bernoulli
RP achlioptas
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
0.42
0.605
0.666
0.143
0.48
0.658
0.789
0.669
0.7
0.79
0.839
0.755
0.724
0.809
0.789
0.456
0.812
0.889
0.887
0.88
0.834
0.885
0.685
Autoencoder
PCA
RP normal
RP bernoulli
RP achlioptas
0.368
0.502
0.62
0.143
0.482
0.677
0.811
0.624
0.719
0.789
0.799
0.758
0.732
0.776
0.822
0.503
0.834
0.859
0.854
0.903
0.936
0.684
0.847
0.684
COF
PCA
RP normal
RP bernoulli
RP achlioptas
0.222
0.185
0.208
0.143
0.211
0.192
0.167
0.212
0.191
0.162
0.184
0.197
0.177
0.19
0.177
0.239
0.174
0.172
0.191
0.173
0.17
0.186
0.167
0.191
COPOD
PCA
RP normal
RP bernoulli
RP achlioptas
0.253
0.257
0.242
0.143
0.292
0.29
0.307
0.284
0.255
0.292
0.329
0.231
0.263
0.338
0.314
0.29
0.238
0.214
0.196
0.24
0.22
0.308
0.215
0.248


DeepSVDD
ECOD
HBOS
iForest
PCA
RP normal
RP bernoulli
RP achlioptas


KNN
LMDD
LODA
LOF
PCA
RP normal
RP bernoulli
RP achlioptas
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1
2
3
4
5
6
0.431 0.622 0.708 0.143
0.475 0.676 0.804 0.67
0.704 0.816 0.842 0.763
0.727 0.812 0.814 0.495
0.818 0.897 0.9 0.912
0.907 0.856 0.891 0.691
0.846 0.857 0.857 0.143
0.364 0.377 0.398 0.294
0.469 0.411 0.369 0.629
0.504 0.318 0.372 0.405
0.493 0.589 0.49 0.528
0.549 0.359 0.29 0.448
0.335 0.637 0.496 0.143
0.452 0.569 0.767 0.621
0.664 0.794 0.79 0.628
0.711 0.733 0.813 0.345
0.827 0.862 0.769 0.769
0.929 0.498 0.843 0.697
0.385 0.41 0.272 0.143
0.474 0.606 0.598 0.633
0.667 0.639 0.767 0.726
0.725 0.81 0.657 0.452
0.799 0.84 0.871 0.807
0.88 0.853 0.758 0.673


OCSVM
PCA
RP normal
RP bernoulli
RP achlioptas
0.636 0.803 0.824 0.857
0.663 0.843 0.882 0.773
0.789 0.9 0.911 0.838
0.792 0.894 0.9 0.691
0.856 0.92 0.921 0.928
0.926 0.912 0.927 0.771
PCA
0.332 0.626 0.393 0.143
0.474 0.682 0.774 0.674
0.72 0.753 0.815 0.751
0.725 0.771 0.793 0.143
0.838 0.824 0.84 0.872
0.929 0.692 0.867 0.143
ROD
0.143 0.143 0.236 0.143
0.372 0.251 0.183 0.381
0.434 0.159 0.165 0.172
0.358 0.362 0.222 0.264
0.281 0.182 0.281 0.208
0.192 0.412 0.275 0.264
SO_GAAL
0.191 0.143 0.143 0.143
0.34 0.143 0.143 0.143
0.458 0.143 0.143 0.324
0.427 0.426 0.476 0.315
0.353 0.441 0.405 0.143
0.439 0.143 0.143 0.399

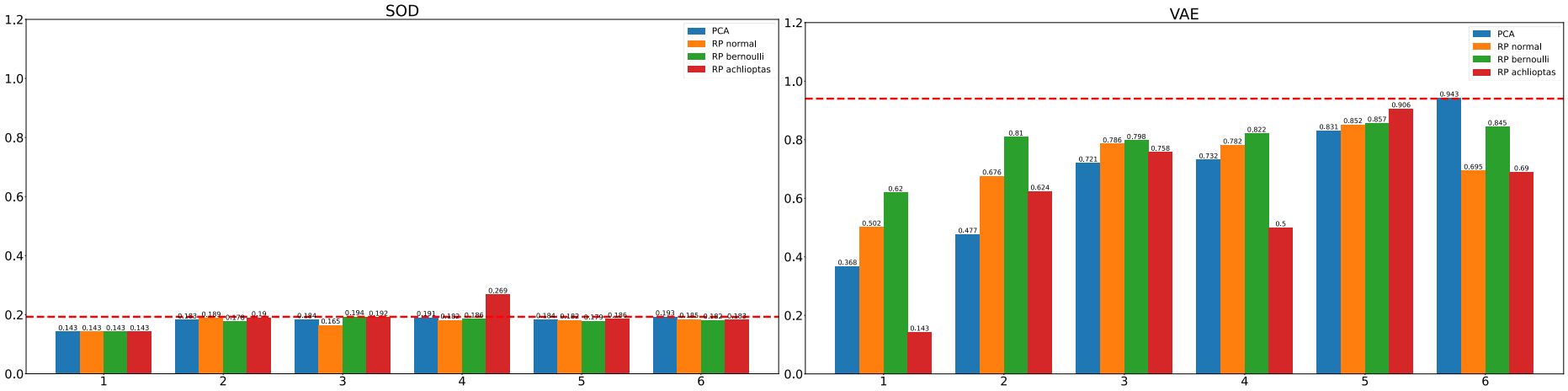
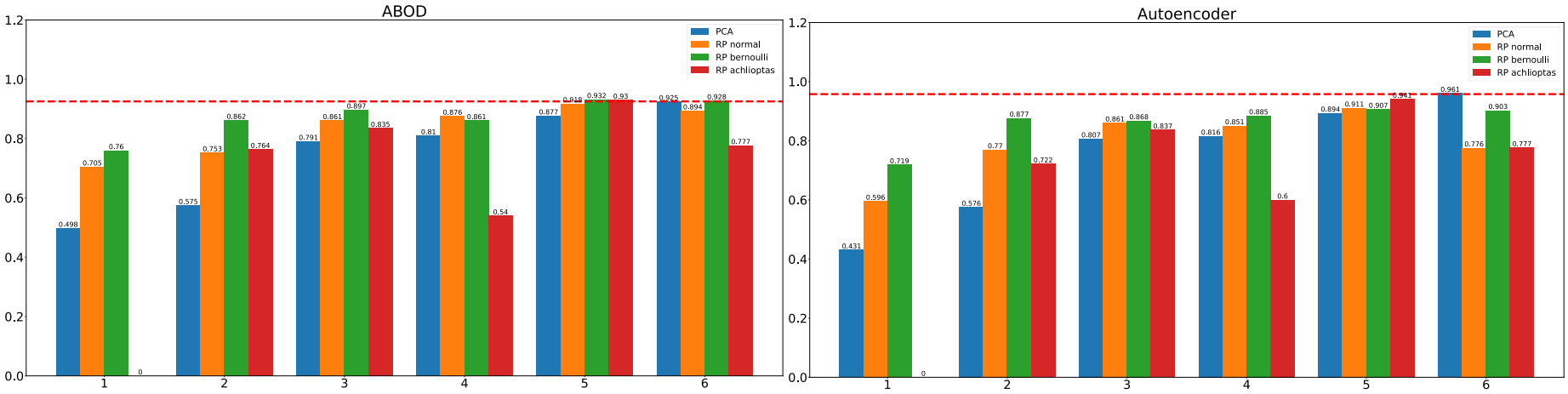
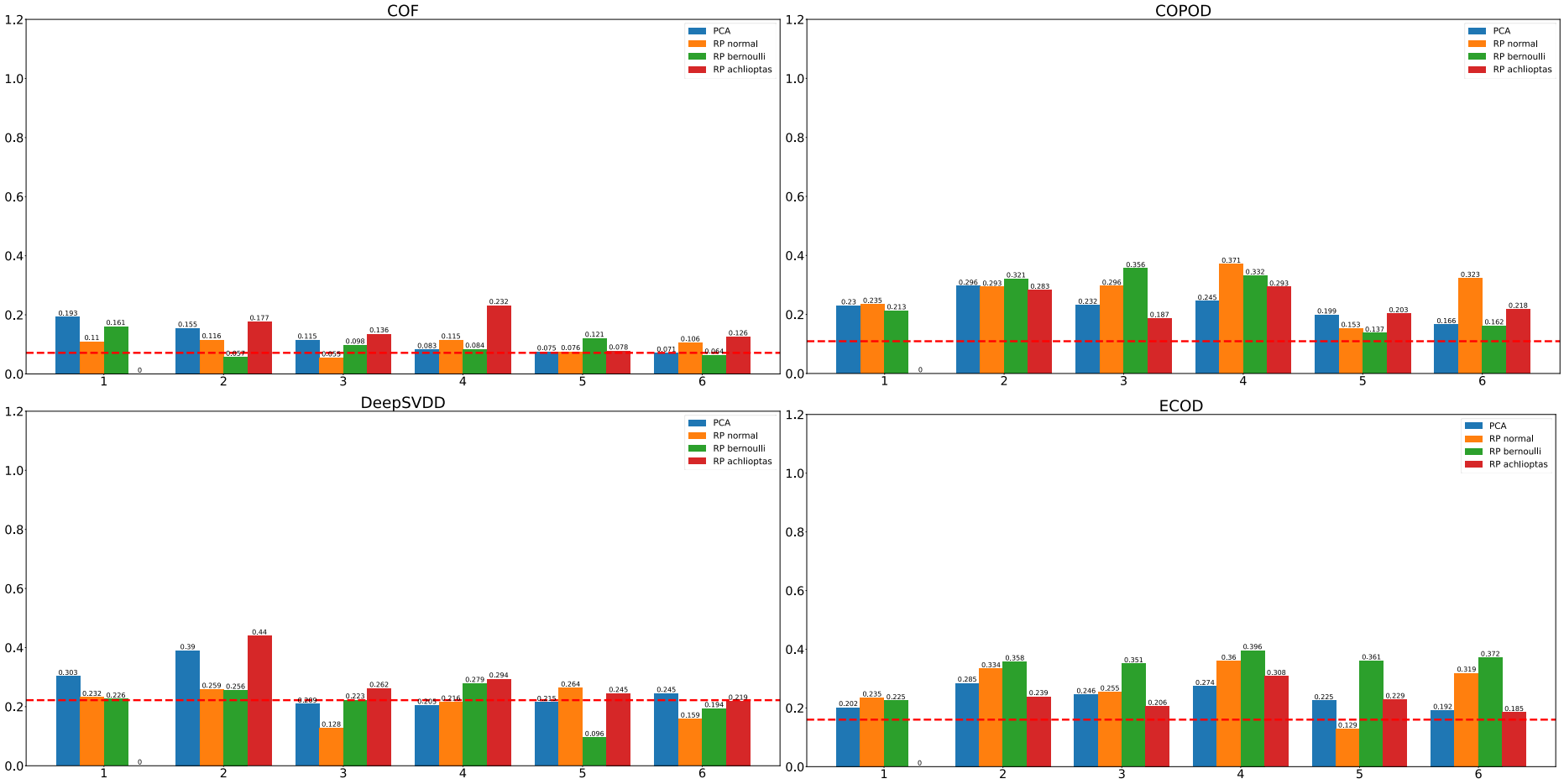
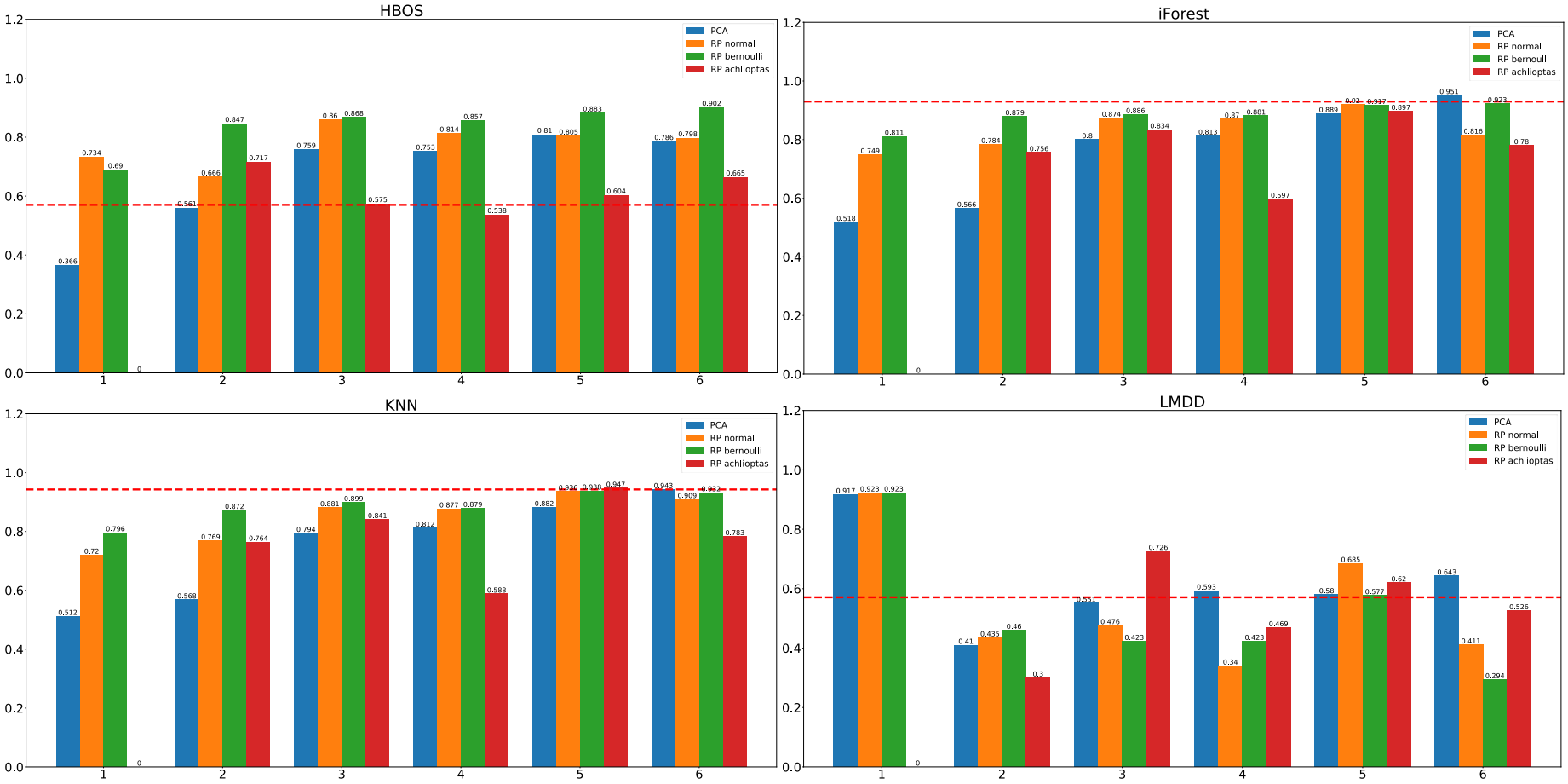
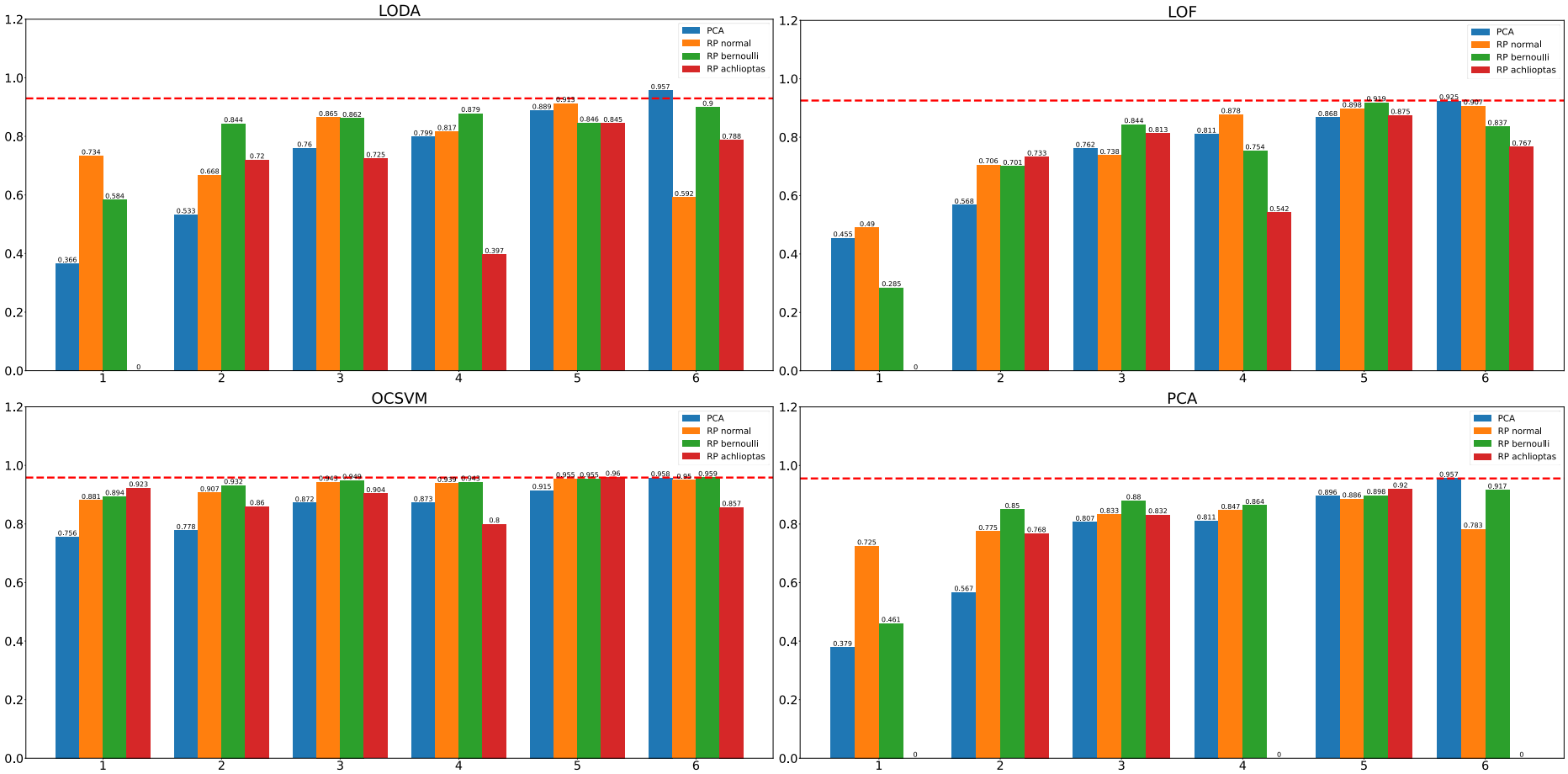


Fig.S1. The Accuracy of ADMs on P4 dataset (10% GE) with different data transformation methods




COF
PCA
RP normal
RP bernoulli
RP achlioptas
0.193 0.11 0.161 0
0.155 0.116 0.059 0.177
0.115 0.054 0.098 0.136
0.083 0.115 0.084 0.232
0.075 0.076 0.121 0.078
0.071 0.106 0.064 0.126
COPOD
0.23 0.235 0.213 0
0.296 0.293 0.321 0.283
0.232 0.296 0.356 0.187
0.245 0.371 0.332 0.293
0.199 0.153 0.137 0.203
0.166 0.323 0.162 0.218
DeepSVDD
0.303 0.232 0.226 0
0.39 0.259 0.256 0.44
0.209 0.128 0.223 0.262
0.205 0.216 0.279 0.294
0.215 0.264 0.096 0.245
0.245 0.159 0.194 0.219
ECOD
0.202 0.235 0.225 0
0.285 0.334 0.358 0.239
0.246 0.255 0.351 0.206
0.274 0.36 0.396 0.308
0.225 0.129 0.361 0.229
0.192 0.319 0.372 0.185


HBOS
PCA
RP normal
RP bernoulli
RP achlioptas
0.366 0.734 0.69 0
0.561 0.666 0.847 0.717
0.759 0.86 0.868 0.575
0.753 0.814 0.857 0.538
0.81 0.805 0.883 0.604
0.786 0.798 0.902 0.665
iForest
0.518 0.749 0.811 0
0.566 0.784 0.879 0.756
0.8 0.874 0.886 0.834
0.813 0.87 0.881 0.597
0.889 0.92 0.917 0.897
0.951 0.816 0.923 0.78
KNN
0.512 0.72 0.796 0
0.568 0.769 0.872 0.764
0.794 0.881 0.899 0.841
0.812 0.877 0.879 0.588
0.882 0.936 0.938 0.947
0.943 0.909 0.932 0.783
LMDD
0.917 0.923 0.923 0
0.41 0.435 0.46 0.3
0.551 0.476 0.423 0.726
0.593 0.34 0.423 0.469
0.58 0.685 0.577 0.62
0.643 0.411 0.294 0.526


LODA
LOF
OCSVM
PCA
PCA
RP normal
RP bernoulli
RP achlioptas
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1
2
3
4
5
6
0.366
0.734
0.584
0
0.533
0.668
0.844
0.72
0.76
0.865
0.862
0.725
0.799
0.817
0.879
0.397
0.889
0.911
0.846
0.845
0.957
0.592
0.9
0.788
0.455
0.49
0.285
0
0.568
0.706
0.701
0.733
0.762
0.738
0.844
0.813
0.811
0.878
0.754
0.542
0.868
0.898
0.919
0.875
0.925
0.907
0.837
0.767
0.756
0.881
0.894
0.923
0.778
0.907
0.932
0.86
0.872
0.943
0.949
0.904
0.873
0.939
0.945
0.8
0.915
0.955
0.955
0.96
0.958
0.95
0.959
0.857
0.379
0.725
0.461
0
0.567
0.775
0.85
0.768
0.807
0.833
0.88
0.832
0.811
0.847
0.864
0
0.896
0.886
0.898
0.92
0.957
0.783
0.917
0

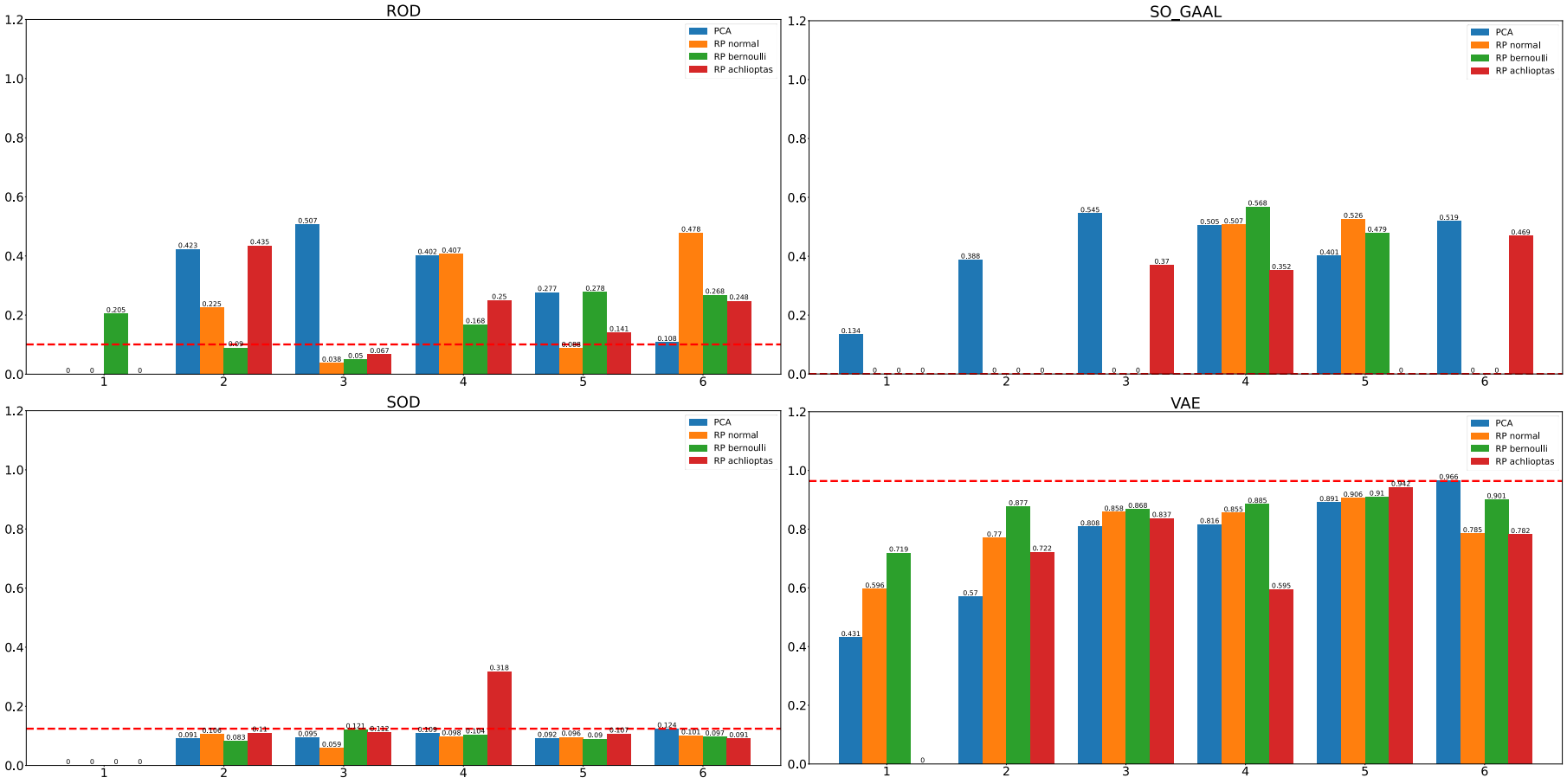
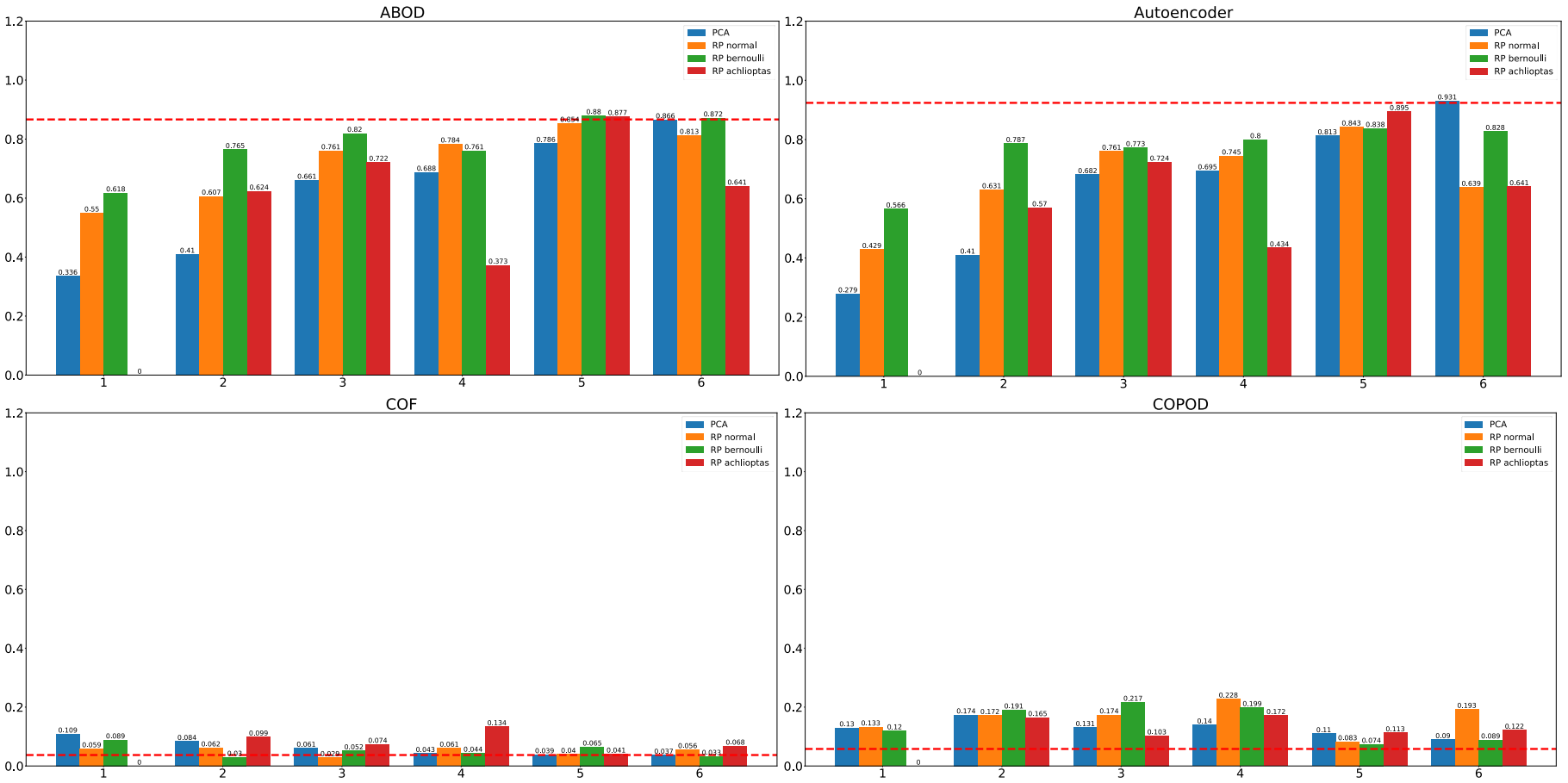
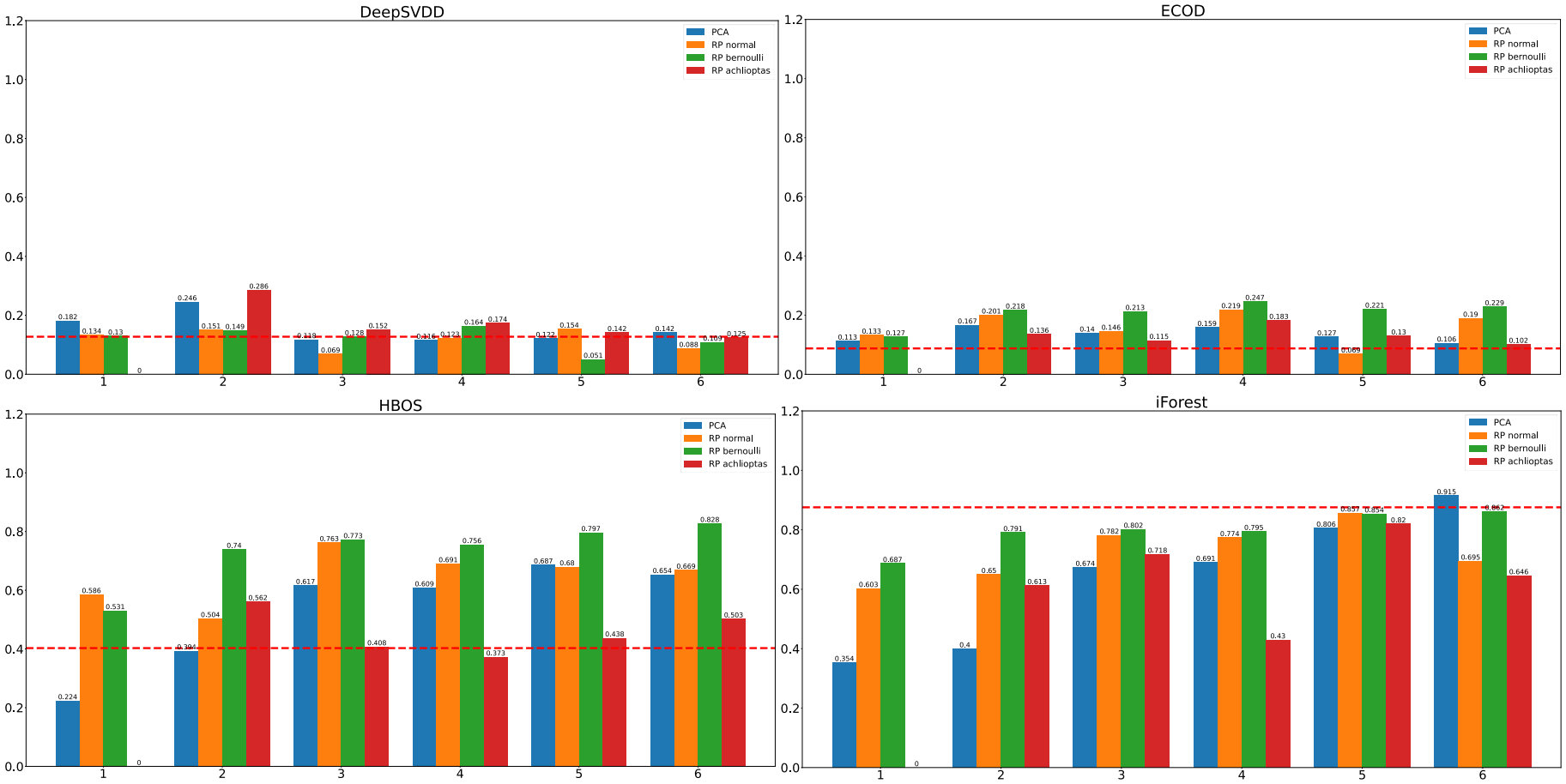
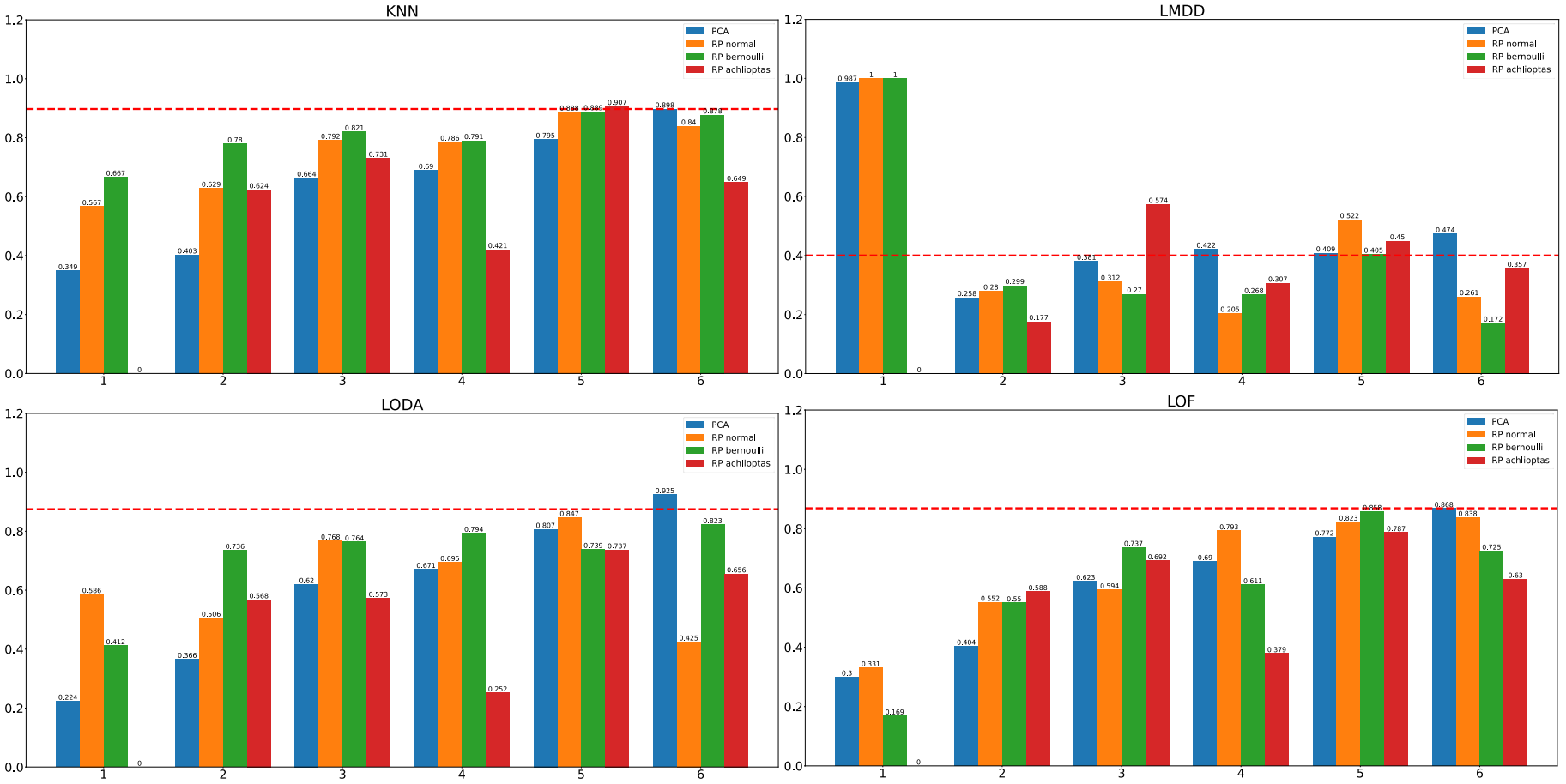
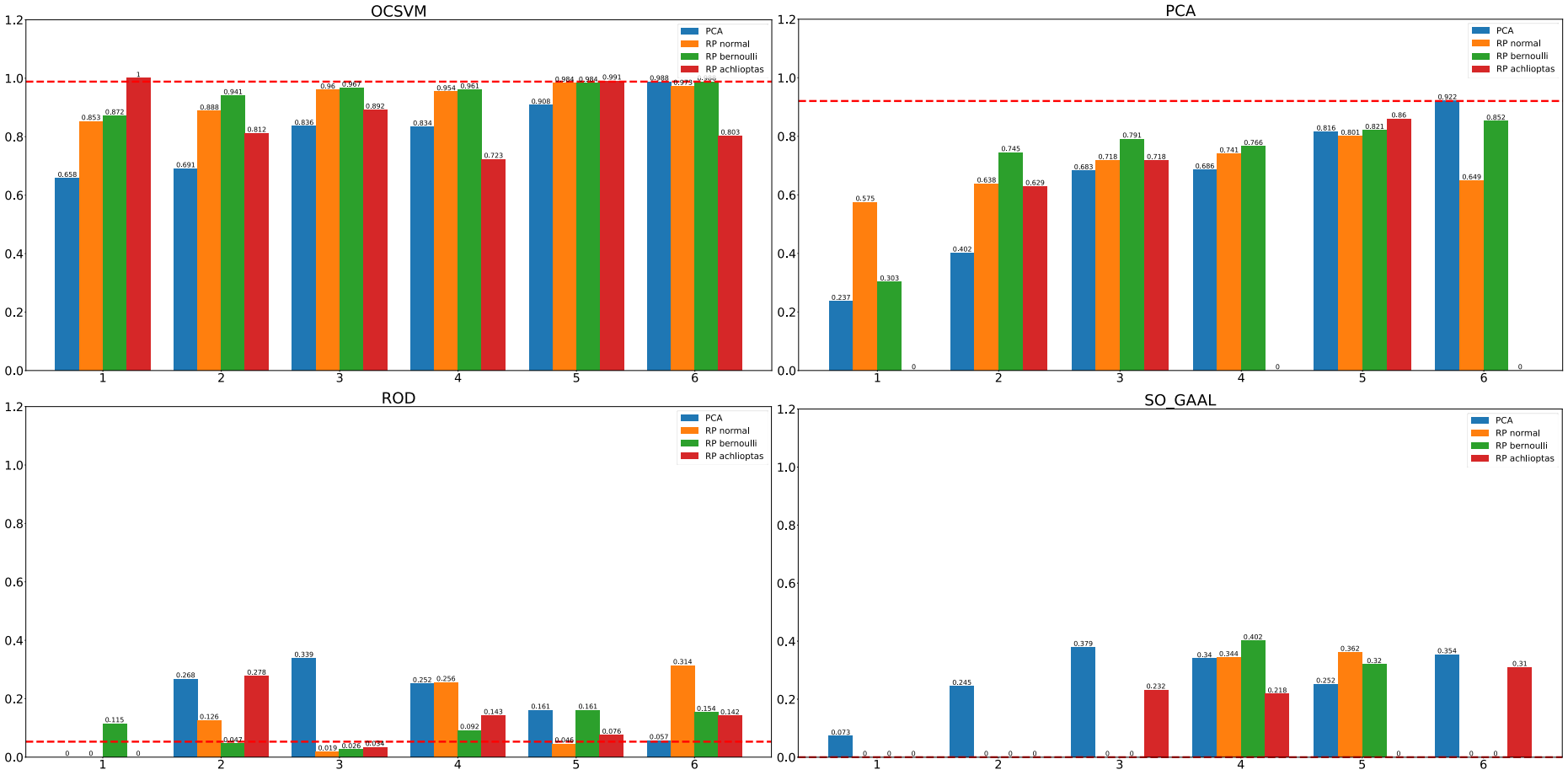


Fig.S2. The F1 Score of ADMs on P4 dataset (10% GE) with different data transformation methods


ABOD
Autoencoder
COF
COPOD
PCA
RP normal
RP bernoulli
RP achlioptas


DeepSVDD
ECOD
HBOS
iForest
PCA
RP normal
RP bernoulli
RP achlioptas


KNN
LMDD
LODA
LOF
PCA
RP normal
RP bernoulli
RP achlioptas


OCSVM
PCA
RP normal
RP bernoulli
RP achlioptas
ROD
SO_GAAL

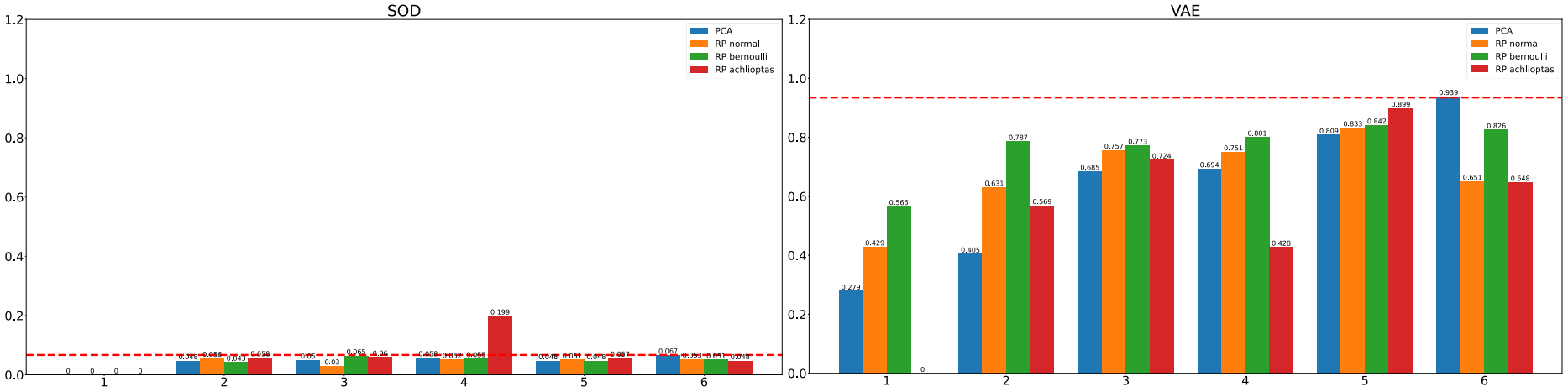
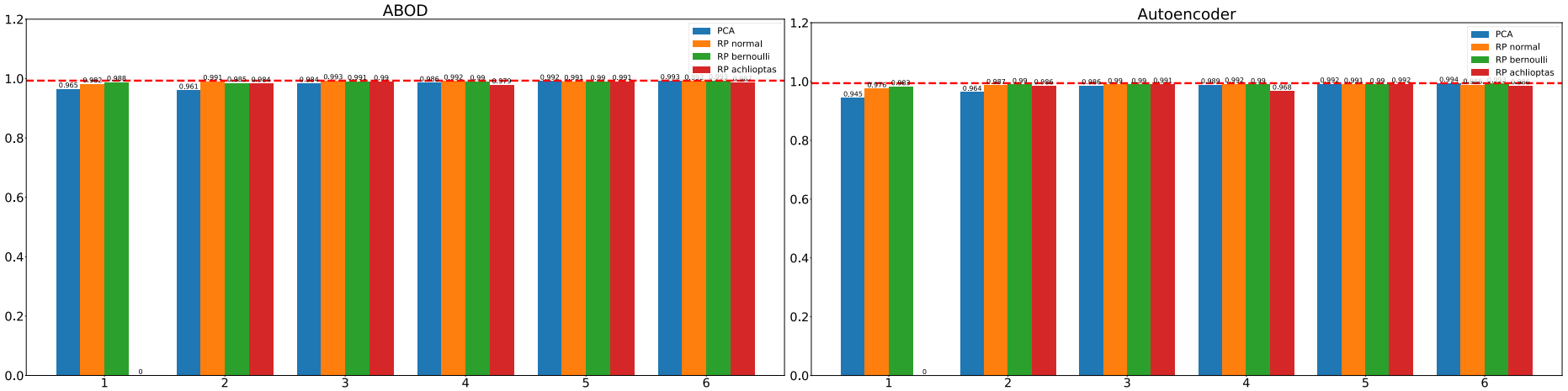
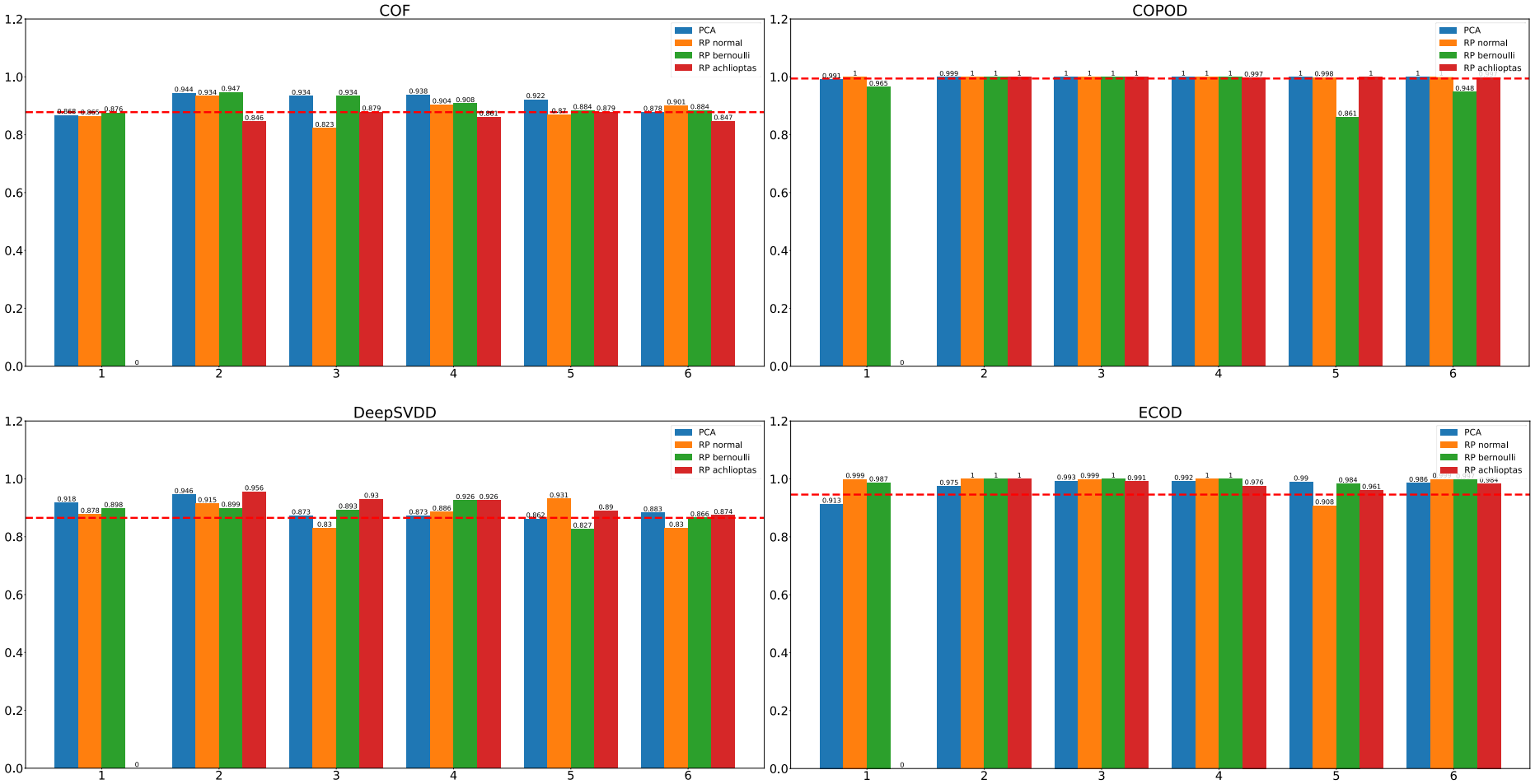
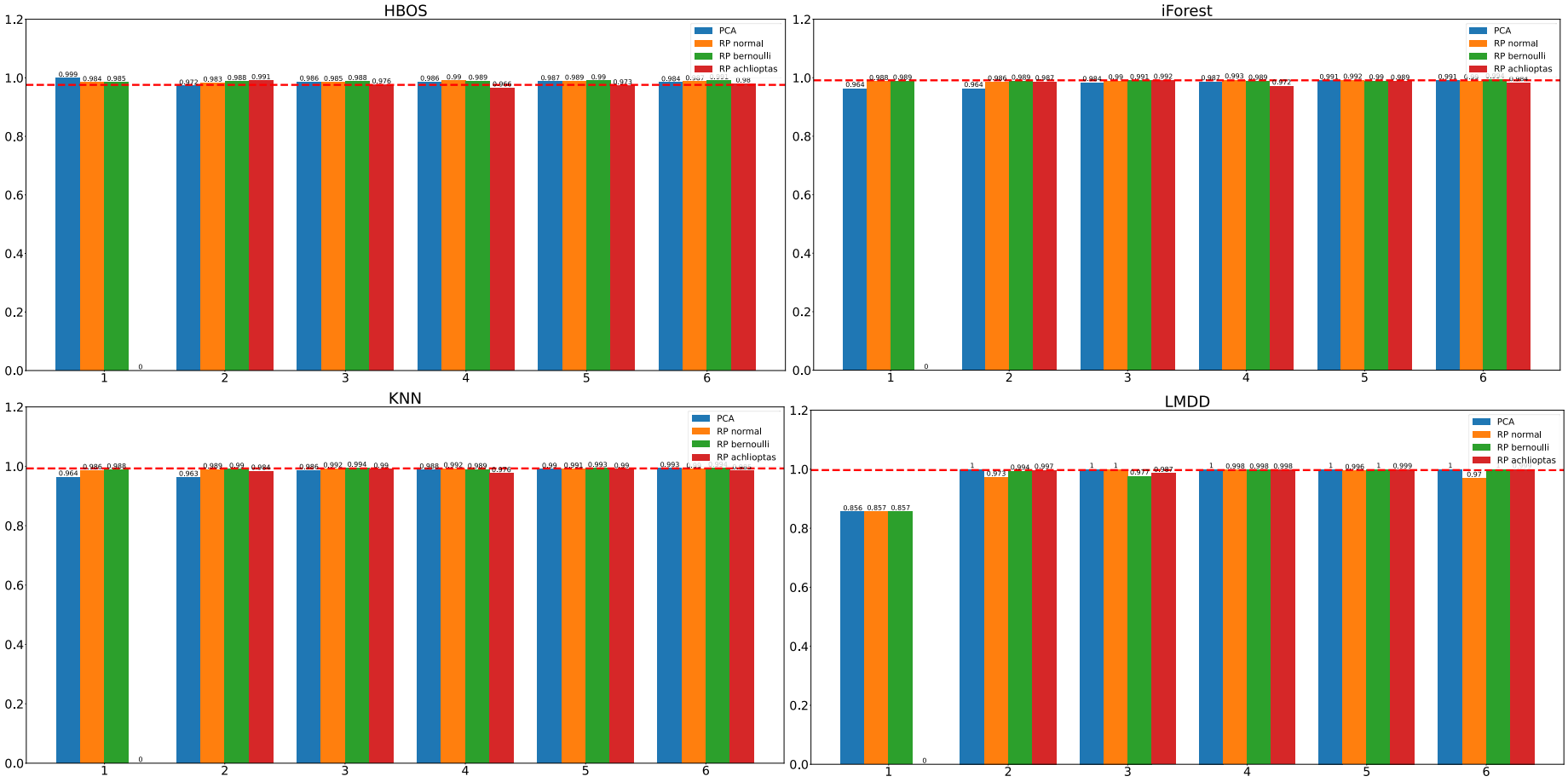
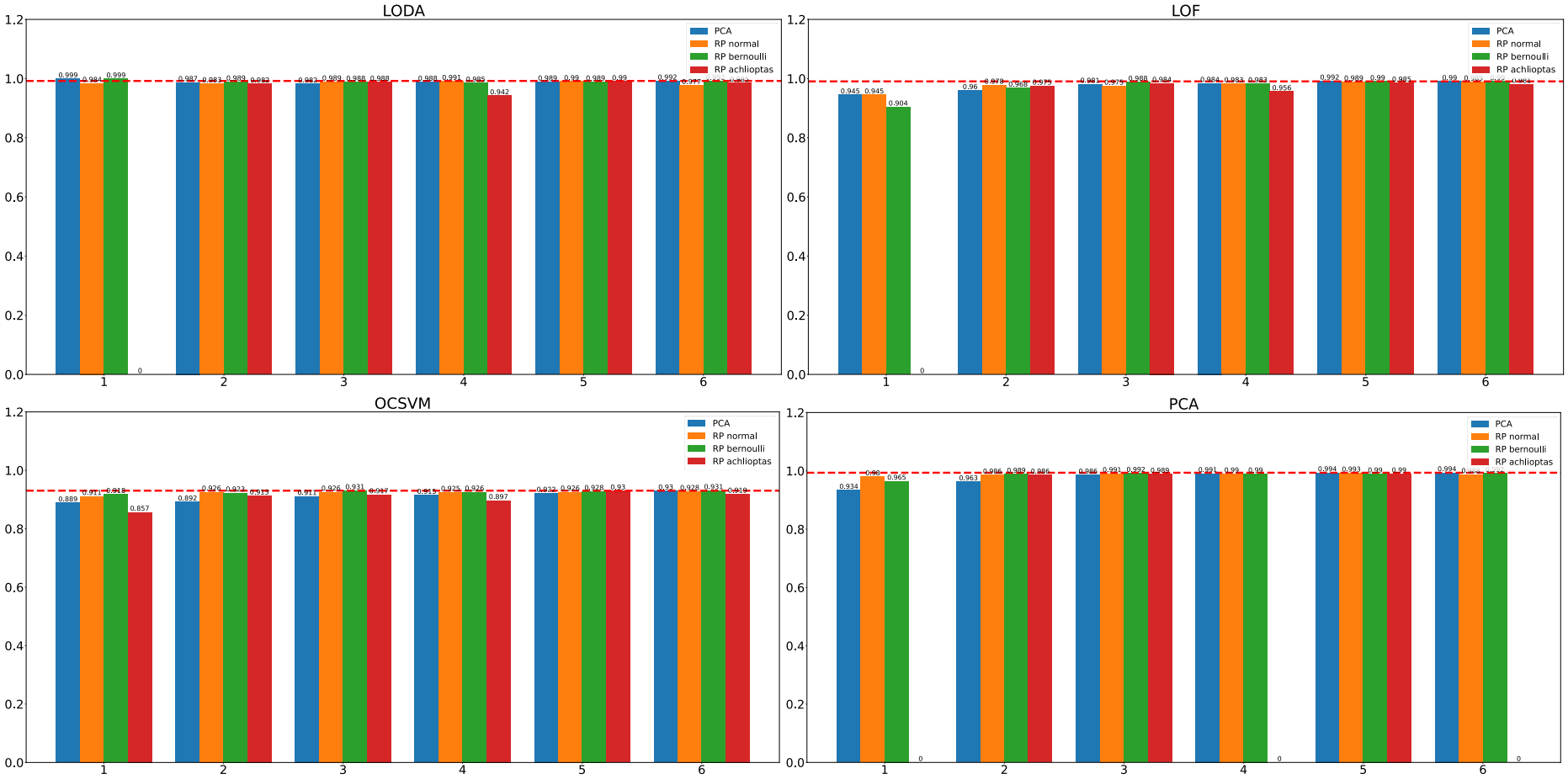


Fig.S3. The OP of ADMs on P4 dataset (10% GE) with different data transformation methods




COF
PCA
RP normal
RP bernoulli
RP achlioptas
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
0.868 0.865 0.876 0
0.944 0.934 0.947 0.846
0.934 0.823 0.934 0.879
0.938 0.904 0.908 0.861
0.922 0.87 0.884 0.879
0.878 0.901 0.884 0.847
COPOD
PCA
RP normal
RP bernoulli
RP achlioptas
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
0.991 1 0.965 0
0.999 1 1 1
1 1 1 1
1 1 1 0.997
1 0.998 0.861 1
1 0.948
DeepSVDD
PCA
RP normal
RP bernoulli
RP achlioptas
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
0.918 0.878 0.898 0
0.946 0.915 0.899 0.956
0.873 0.83 0.893 0.93
0.873 0.886 0.926 0.926
0.862 0.931 0.827 0.89
0.883 0.83 0.866 0.874
ECOD
PCA
RP normal
RP bernoulli
RP achlioptas
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
0.913 0.999 0.987 0
0.975 1 1 1
0.993 0.999 1 0.991
0.992 1 1 0.976
0.99 0.908 0.984 0.961
0.986 0.964


HBOS
iForest
KNN
LMDD
PCA
RP normal
RP bernoulli
RP achlioptas
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1
2
3
4
5
6


LODA
LOF
OCSVM
PCA
PCA
RP normal
RP bernoulli
RP achlioptas
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1
2
3
4
5
6

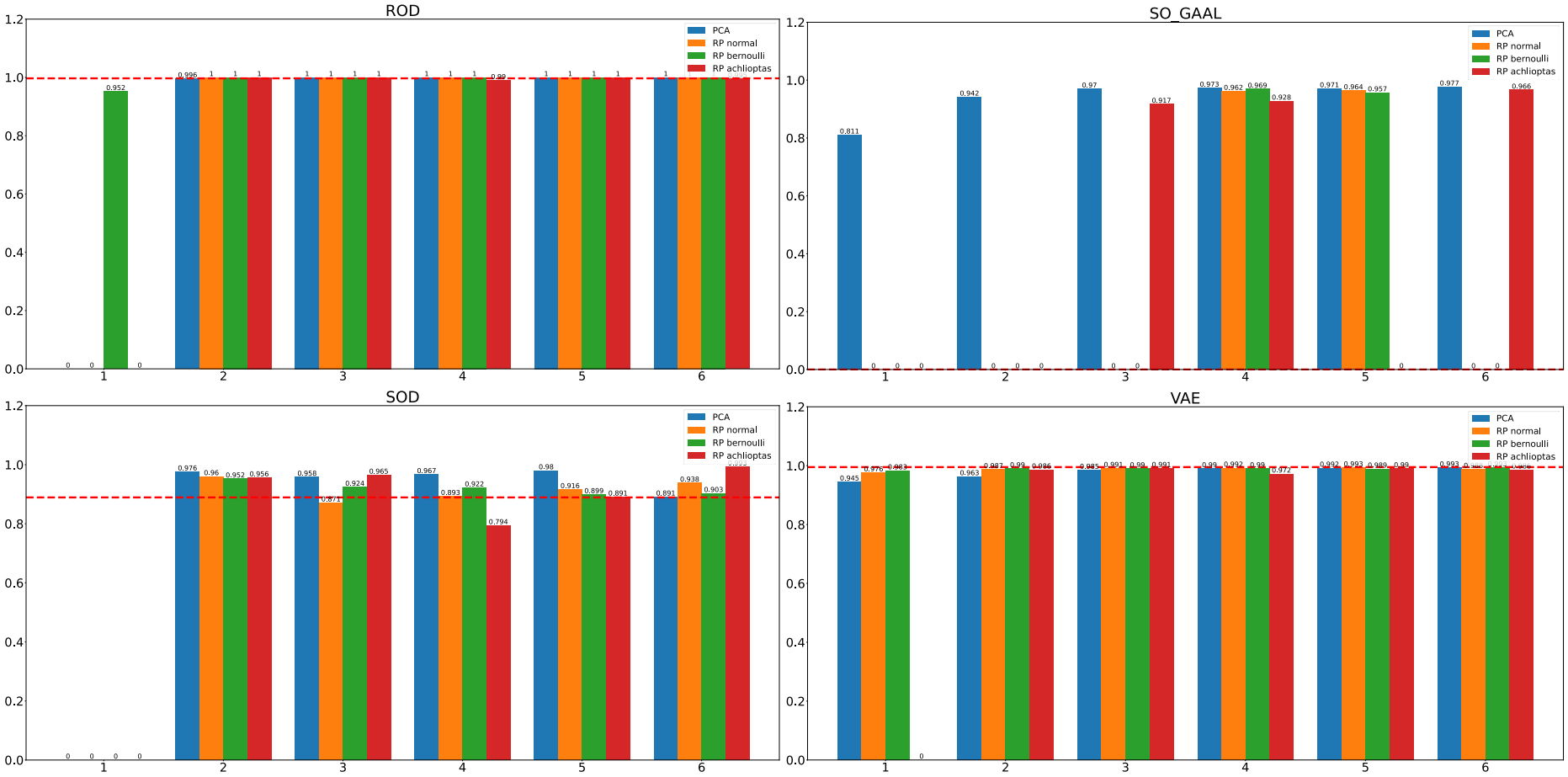


Fig.S4. The Selectivity of ADMs on P4 dataset (10% GE) with different data transformation methods

Table.S5. Accuracy of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±5% of GEs

	iForest	OCSVM	Elliptic Envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.499	0.687	0.491	0.497	0.472	0.458	0.341	0.341	0.341	0.341	0.275
P2	0.430	0.788	0.572	0.561	0.521	0.497	0.363	0.520	0.419	0.520	0.424
P3	0.421	0.756	0.804	0.517	0.784	0.778	0.385	0.521	0.458	0.521	0.903
P4	0.475	0.838	0.646	0.637	0.609	0.561	0.587	0.738	0.684	0.738	0.562
P5	0.307	0.729	0.368	0.376	0.337	0.352	0.747	0.818	0.790	0.818	0.320
P6	0.213	0.514	0.209	0.210	0.206	0.206	0.207	0.354	0.279	0.354	0.153
P7	0.243	0.683	0.357	0.381	0.303	0.313	0.279	0.556	0.464	0.556	0.332
P8	0.259	0.917	0.246	0.232	0.243	0.221	0.173	0.366	0.312	0.366	0.216
P9	0.214	0.923	0.255	0.174	0.226	0.216	0.227	0.497	0.424	0.497	0.270
P10	0.248	0.685	0.351	0.344	0.228	0.210	0.421	0.731	0.649	0.731	0.581
P11	0.199	0.923	0.184	0.217	0.184	0.187	0.183	0.514	0.429	0.514	0.245
P12	0.265	0.786	0.635	0.569	0.593	0.605	0.547	0.758	0.710	0.758	0.740
P13	0.181	0.578	0.167	0.178	0.159	0.165	0.104	0.441	0.315	0.441	0.156
P14	0.328	0.947	0.256	0.291	0.085	0.083	0.378	0.815	0.737	0.815	0.782
P15	0.248	0.691	0.082	0.164	0.071	0.070	0.137	0.664	0.509	0.664	0.344
P16	0.355	0.980	0.033	0.077	0.020	0.020	0.567	0.944	0.896	0.944	0.980
Average	0.305	0.777	0.354	0.339	0.315	0.309	0.353	0.597	0.526	0.599	0.455

Table.S6. Accuracy of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±10% of GEs

	iForest	OCSVM	Elliptic Envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.866	0.849	0.861	0.867	0.830	0.821	0.545	0.545	0.545	0.545	0.569
P2	0.814	0.902	0.885	0.878	0.896	0.897	0.664	0.744	0.691	0.744	0.834
P3	0.670	0.845	0.997	0.775	0.998	0.998	0.566	0.660	0.615	0.660	0.997
P4	0.887	0.926	0.916	0.907	0.940	0.927	0.874	0.924	0.921	0.924	0.897
P5	0.628	0.888	0.738	0.688	0.756	0.763	0.847	0.875	0.861	0.875	0.734
P6	0.245	0.573	0.258	0.246	0.259	0.248	0.210	0.385	0.306	0.385	0.174
P7	0.365	0.760	0.569	0.569	0.566	0.563	0.496	0.698	0.628	0.698	0.559
P8	0.524	0.917	0.638	0.446	0.626	0.592	0.354	0.572	0.505	0.572	0.680
P9	0.349	0.923	0.514	0.360	0.520	0.505	0.420	0.641	0.586	0.641	0.543
P10	0.456	0.871	0.765	0.689	0.787	0.773	0.717	0.902	0.877	0.902	0.882
P11	0.372	0.923	0.558	0.366	0.550	0.563	0.426	0.767	0.690	0.767	0.724
P12	0.459	0.882	0.920	0.767	0.959	0.956	0.768	0.879	0.860	0.879	0.958
P13	0.266	0.658	0.279	0.280	0.273	0.277	0.148	0.564	0.406	0.564	0.307
P14	0.482	0.952	0.762	0.634	0.758	0.754	0.727	0.915	0.852	0.915	0.928
P15	0.368	0.813	0.366	0.406	0.347	0.344	0.330	0.794	0.672	0.794	0.874
P16	0.525	0.980	0.039	0.278	0.020	0.020	0.813	0.956	0.931	0.956	0.989
Average	0.517	0.854	0.629	0.572	0.630	0.625	0.557	0.739	0.684	0.739	0.728

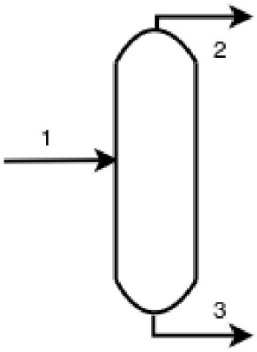
Table.S7. OP of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±5% of GEs

	iForest	OCSVM	Elliptic Envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.365	0.739	0.349	0.363	0.326	0.307	0.136	0.136	0.136	0.136	0.034
P2	0.345	0.830	0.511	0.500	0.450	0.423	0.262	0.461	0.336	0.461	0.329
P3	0.377	0.773	0.790	0.483	0.767	0.762	0.337	0.496	0.424	0.496	0.898
P4	0.394	0.880	0.594	0.583	0.550	0.494	0.527	0.734	0.660	0.734	0.491
P5	0.227	0.757	0.295	0.305	0.259	0.278	0.791	0.899	0.858	0.899	0.237
P6	0.115	0.514	0.109	0.110	0.107	0.105	0.108	0.303	0.205	0.303	0.036
P7	0.177	0.698	0.300	0.326	0.242	0.253	0.213	0.545	0.436	0.545	0.269
P8	0.200	1.000	0.184	0.169	0.180	0.155	0.102	0.328	0.265	0.328	0.148
P9	0.154	1.000	0.197	0.111	0.165	0.154	0.172	0.486	0.402	0.486	0.213
P10	0.193	0.697	0.302	0.296	0.169	0.150	0.384	0.746	0.652	0.746	0.554
P11	0.137	1.000	0.119	0.159	0.119	0.122	0.119	0.504	0.404	0.504	0.187
P12	0.207	0.807	0.606	0.535	0.561	0.575	0.513	0.765	0.707	0.765	0.722
P13	0.136	0.585	0.120	0.133	0.111	0.118	0.050	0.430	0.288	0.430	0.108
P14	0.304	0.984	0.226	0.263	0.047	0.045	0.353	0.828	0.745	0.828	0.778
P15	0.226	0.700	0.050	0.137	0.038	0.037	0.109	0.673	0.506	0.673	0.325
P16	0.345	1.000	0.014	0.059	0.000	0.000	0.559	0.960	0.907	0.960	0.984
Average	0.2439	0.8103	0.2979	0.2833	0.2557	0.2486	0.2959	0.5809	0.4957	0.5809	0.395

Table.S8. OP of the top 6 ADMs and 5 statistical tests on the 16 datasets with ±10% of GEs

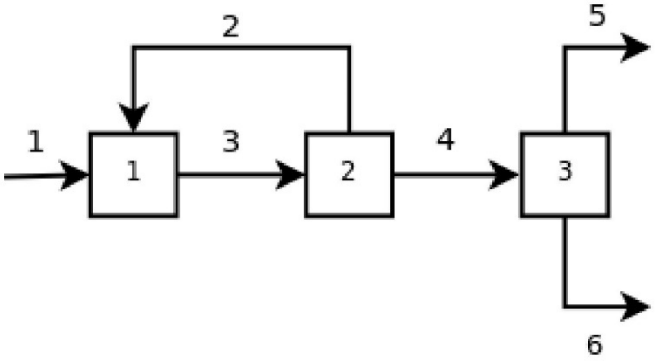
	iForest	OCSVM	Elliptic Envelope	KNN	VAE	PCA	GT	MT	NT	GLR	IQR
P1	0.858	0.967	0.849	0.858	0.810	0.798	0.407	0.407	0.407	0.407	0.426
P2	0.794	0.966	0.877	0.869	0.889	0.890	0.614	0.722	0.655	0.722	0.808
P3	0.647	0.870	1.000	0.762	1.000	1.000	0.534	0.645	0.592	0.645	1.000
P4	0.876	0.988	0.910	0.898	0.935	0.921	0.862	0.949	0.936	0.949	0.880
P5	0.590	0.935	0.712	0.656	0.732	0.739	0.901	0.958	0.934	0.958	0.703
P6	0.152	0.582	0.165	0.153	0.167	0.153	0.108	0.339	0.233	0.339	0.060
P7	0.309	0.785	0.534	0.534	0.530	0.525	0.451	0.704	0.616	0.704	0.519
P8	0.489	1.000	0.612	0.402	0.599	0.561	0.301	0.553	0.476	0.553	0.654
P9	0.302	1.000	0.479	0.314	0.486	0.470	0.383	0.643	0.580	0.643	0.509
P10	0.418	0.896	0.748	0.667	0.771	0.757	0.705	0.933	0.899	0.933	0.878
P11	0.325	1.000	0.524	0.320	0.515	0.530	0.381	0.777	0.687	0.777	0.704
P12	0.418	0.912	0.915	0.750	0.958	0.955	0.753	0.898	0.870	0.898	0.960
P13	0.226	0.669	0.239	0.241	0.232	0.236	0.097	0.563	0.386	0.563	0.268
P14	0.464	0.988	0.752	0.620	0.748	0.744	0.718	0.932	0.865	0.932	0.930
P15	0.350	0.827	0.344	0.387	0.324	0.321	0.307	0.807	0.675	0.807	0.873
P16	0.519	1.000	0.020	0.264	0.000	0.000	0.810	0.971	0.944	0.971	0.995
Average	0.4836	0.8991	0.6050	0.5434	0.6060	0.6000	0.5208	0.7376	0.6722	0.7376	0.698

Fig.S5. Separator equipment used in GED of Problem 1 (P1)



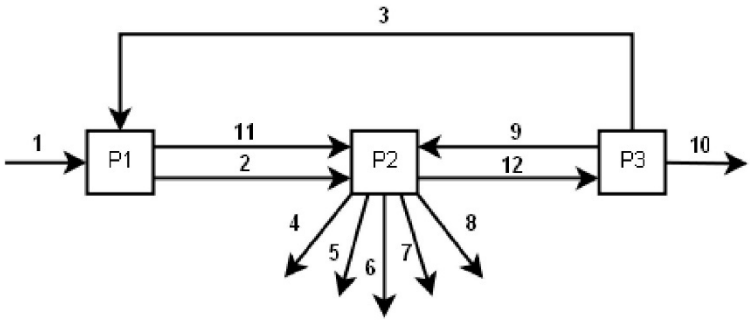
Stream Name	Real Value	Uncertainty
1	8.5	3.2541
2	4.5	3.2200
3	4	2.4150

Fig.S6. Reaction and separation flowsheet used in GED of Problem 2 (P2)



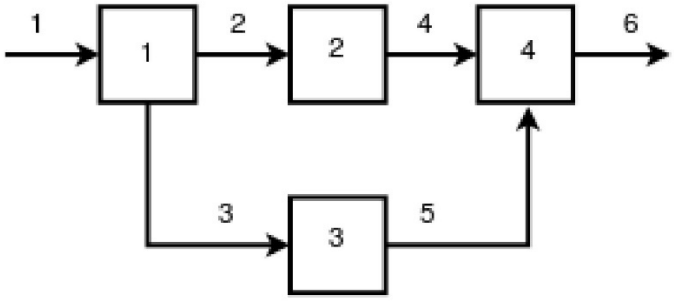
Stream Name	Real Value	Uncertainty
1	11	1.6262
2	10	1.6125
3	21	1.6496
4	11	1.6514
5	4	5.7009
6	4	3.0619

Fig.S7. Atmospheric tower flowsheet used in GED of Problem 3 (P3)



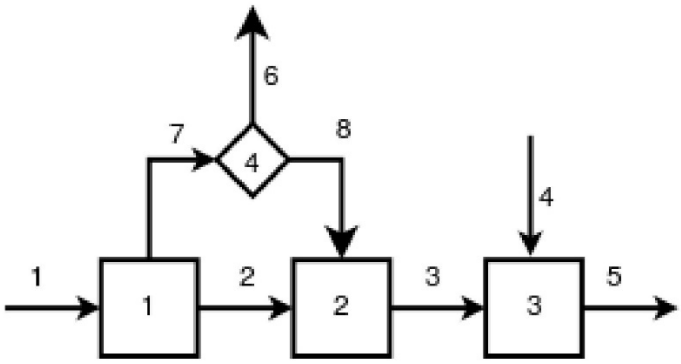
Stream Name	Real Value	Uncertainty
1	189.98	1.1271
2	174.6	1.1159
3	3.139	1.0863
4	32.77	1.0531
5	33.47	1.1954
6	7.25	1.1972
7	0.316	1.1263
8	92.376	1.1468
9	28.629	1.2599
10	23.8	1.2609
11	18.526	1.0802
12	55.568	1.1613

Fig.S8. Heat exchanger with by-pass valve flowsheet used in GED of Problem 4 (P4)



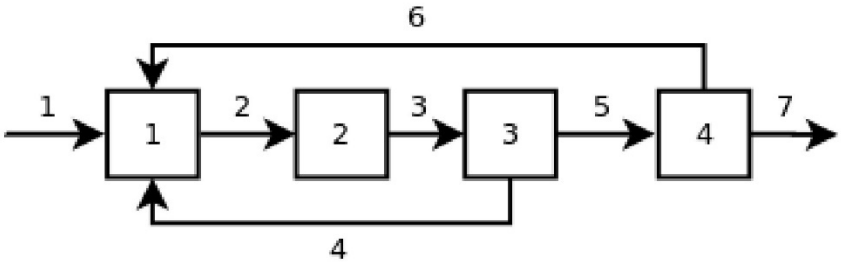
Stream Name	Real Value	Uncertainty
1	100	1.0000
2	64	1.5625
3	36	2.7778
4	64	1.5625
5	36	2.7778
6	100	1.0000

Fig.S9. Generic mass balance flowsheet used in GED of Problem 5 (P5)



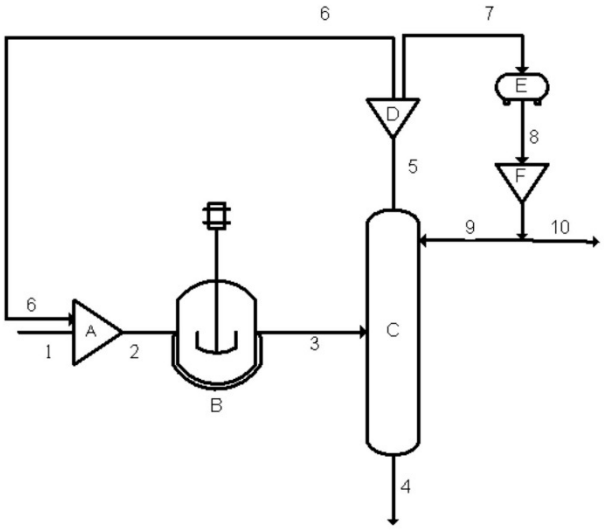
Stream Name	Real Value	Uncertainty
1	98.7	1.0066
2	41.1	1.5598
3	78.9	1.1258
4	30.2	1.8197
5	109.1	0.9574
6	19.8	2.2473
7	57.6	1.3176
8	37.8	1.6265

Fig.S10. Heat exchanger network with recycle flowsheet used in GED of Problem 6 (P6)



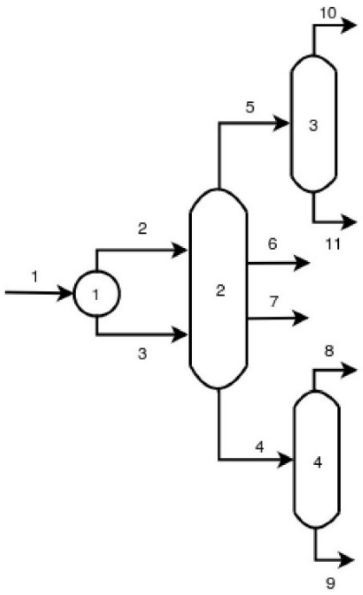
Stream Name	Real Value	Uncertainty
1	5	20.0000
2	15	6.6667
3	15	6.6667
4	5	20.0000
5	10	10.0000
6	5	20.0000
7	5	20.0000

Fig.S11. Diagram of a reaction-separation flowsheet used in GED of Problem 7 (P7)



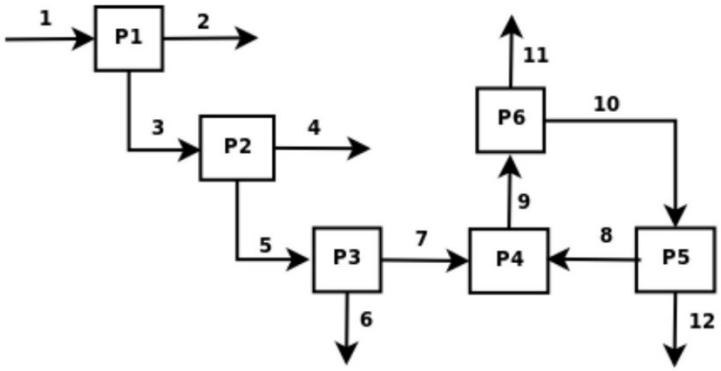
Stream Name	Real Value	Uncertainty
1	50	2.0000
2	75	1.3333
3	75	1.3333
4	48	2.0833
5	30	3.3333
6	25	4.0000
7	5	7.7460
8	5	7.7460
9	3	10.5409
10	2	15.8114

Fig.S12. Juice extraction plant flowsheet used in GED of Problem 8 (P8)



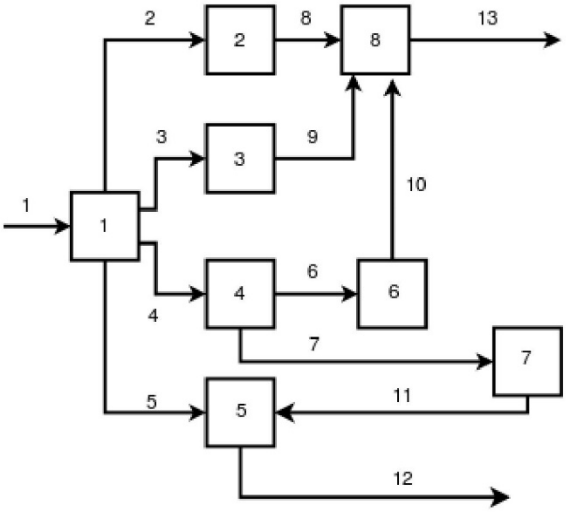
Stream Name	Real Value	Uncertainty
1	3600	3.0892
2	1850	3.0811
3	1750	3.0977
4	2837	3.0772
5	730	3.0288
6	25	3.1200
7	8	2.8500
8	137	2.2993
9	2700	3.1467
10	58	2.9483
11	672	2.9821

Fig.S13. Generic mass balance flowsheet used in GED of Problem 9 (P9)



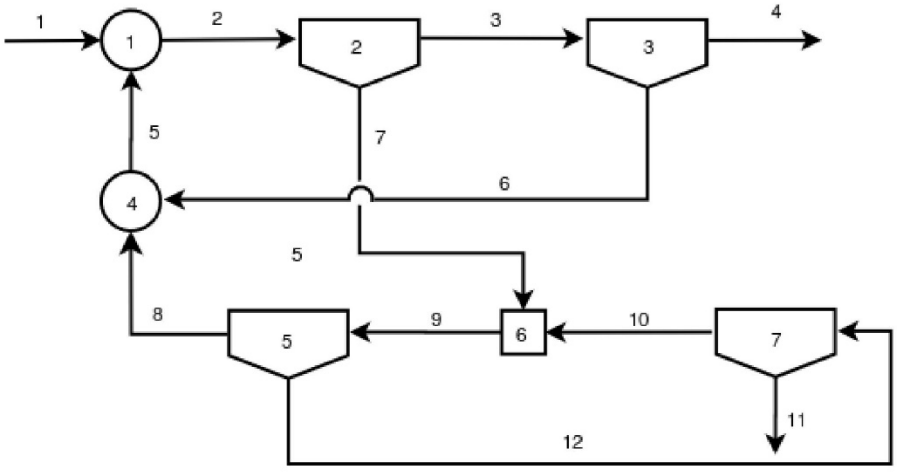
Stream Name	Real Value	Uncertainty
1	230	16.3370
2	21	5.1429
3	209	2.3923
4	35	5.2143
5	174	1.1494
6	15	5.8667
7	159	4.5566
8	50	2.0000
9	209	2.3923
10	94	2.1277
11	115	15.7391
12	44	5.4205

Fig.S14. Generic mass balance flowsheet used in GED of Problem 10 (P10)



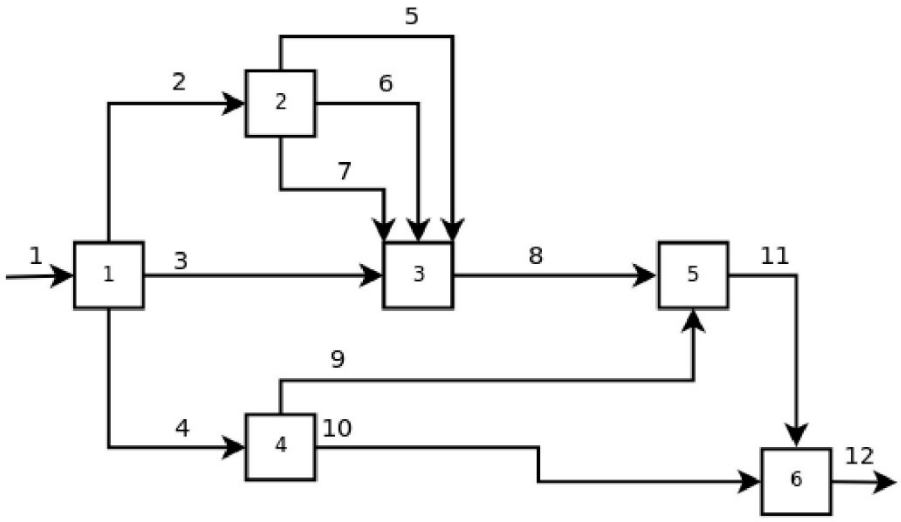
Stream Name	Real Value	Uncertainty
1	28	0.9821
2	5	0.9996
3	5	3.4496
4	7	2.0743
5	11	3.3825
6	4	3.1768
7	3	4.5407
8	5	0.9081
9	5	1.9045
10	4	1.8130
11	3	2.1179
12	14	1.0500
13	14	0.9331

Fig.S15. Mineral beneficiation circuit flowsheet used in GED of Problem 11 (P11)



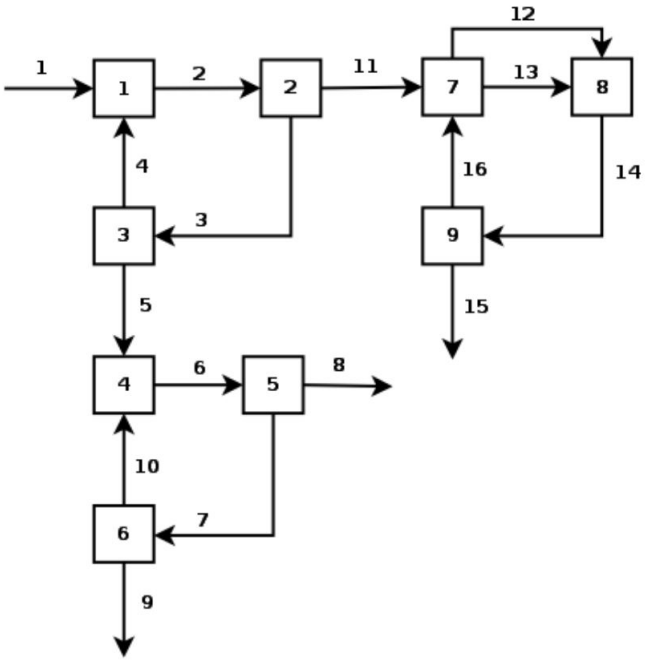
Stream Name	Real Value	Uncertainty
1	690	3.0073
2	725	3.0105
3	700	2.9973
4	685	3.0094
5	35	3.0746
6	15	2.5020
7	25	3.3456
8	20	3.5040
9	30	2.2670
10	5	2.8740
11	5	2.7120
12	10	2.7930

Fig.S16. Generic mass balance data flowsheet used in GED of Problem 12 (P12)



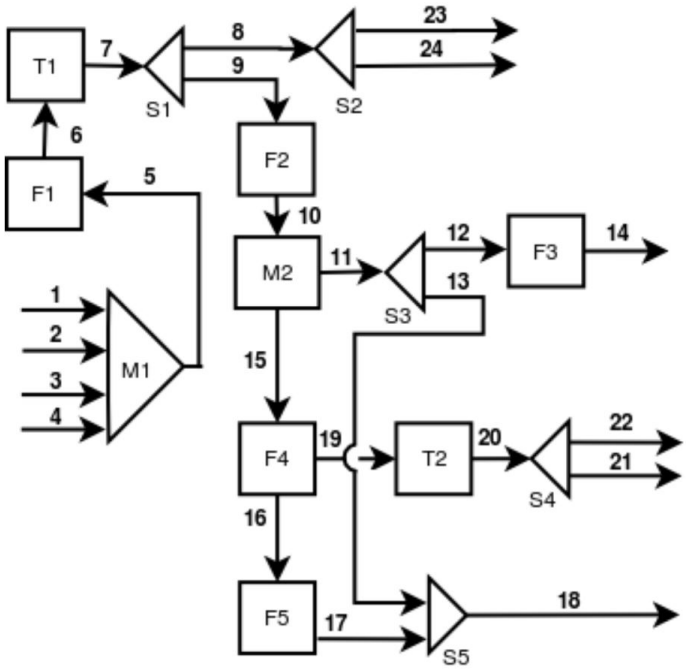
Stream Name	Real Value	Uncertainty
1	100	1.0000
2	30	1.0000
3	40	1.0000
4	30	1.0000
5	15	2.0000
6	5	2.0000
7	10	2.0000
8	70	1.0000
9	10	3.0000
10	20	2.0000
11	80	1.0000
12	100	1.0000

Fig.S17. Flotation circuit flowsheet used in GED of Problem 13 (P13)



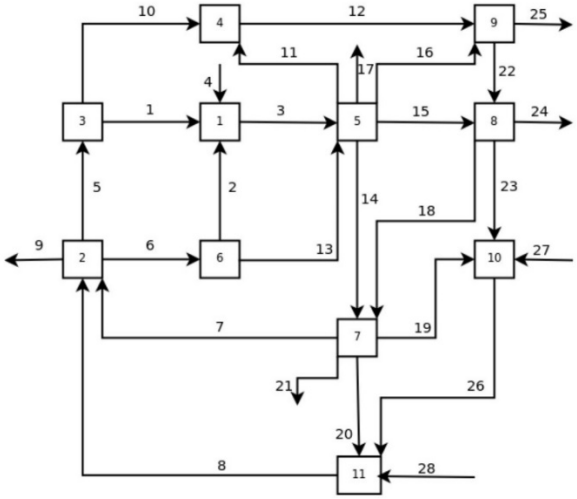
Stream Name	Real Value	Uncertainty
1	25	4.0000
2	27	4.9074
3	22	6.6364
4	2	10.0000
5	20	4.5800
6	24	4.5875
7	14	7.4286
8	10	4.7200
9	10	4.0100
10	4	5.1750
11	5	6.0000
12	7	4.6857
13	1	5.2000
14	8	4.6125
15	5	5.0000
16	3	12.8333

Fig.S18. Proposed mass balance of an industrial water treatment unit flowsheet used in GED of Problem 14 (P14)



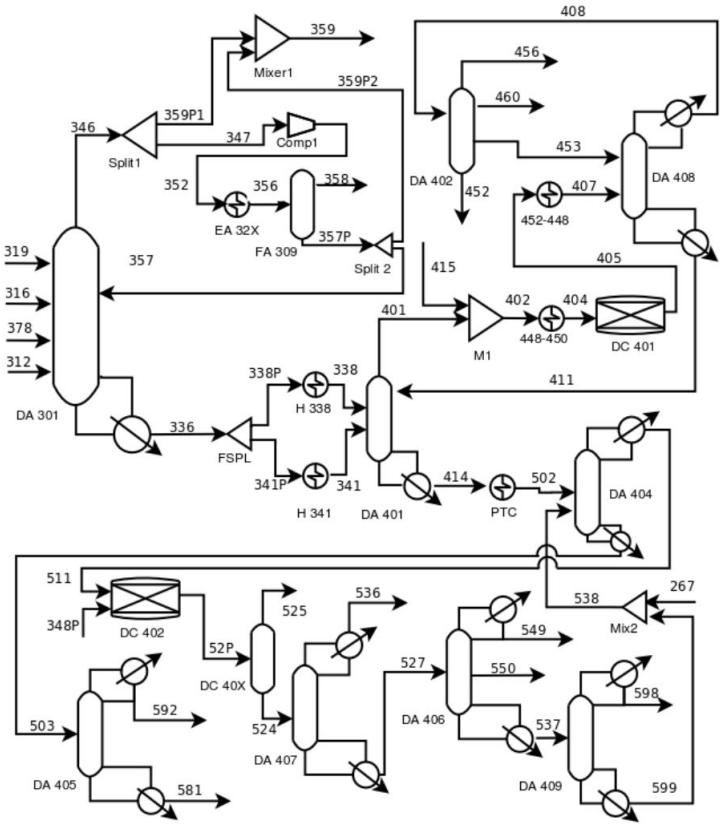
Stream Name	Real Value	Uncertainty
1	50	1.0662
2	150	0.9735
3	140	1.1004
4	140	1.0367
5	480	1.5857
6	480	1.5857
7	480	1.7364
8	220	0.9711
9	260	1.1017
10	260	1.1017
11	85.8	0.9640
12	15.8	2.7911
13	70	1.1029
14	15.8	2.7911
15	174.2	1.1701
16	3.484	2.7191
17	3.484	2.7191
18	73.484	3.8808
19	170.716	1.0884
20	170.716	1.1711
21	55	0.9675
22	115.716	1.2678
23	200	0.9883
24	20	0.7987

Fig.S19. Steam metering system flowsheet used in GED of Problem 15 (P15)



Stream Name	Real Value	Uncertainty
1	0.86	2.5000
2	1	2.5000
3	111.82	2.5000
4	109.96	2.5000
5	53.27	2.5000
6	112.27	2.5000
7	2.32	2.5000
8	164.05	2.5000
9	0.83	2.5000
10	52.41	2.5000
11	14.86	2.5000
12	67.27	2.5000
13	111.27	2.5000
14	91.86	2.5000
15	60	2.5000
16	23.64	2.5000
17	32.73	2.5000
18	16.23	2.5000
19	7.85	2.5000
20	10.5	2.5000
21	87.32	2.5000
22	5.45	2.5000
23	2.59	2.5000
24	46.63	2.5000
25	85.46	2.5000
26	81.32	2.5000
27	70.79	2.5000
28	72.23	2.5000

Fig.S20. Ethylene plant flowsheet used in GED of Problem 16 (P16)



Stream Name	Real Value	Uncertainty
S319	225.45	0.9159
S316	167.89	0.7829
S312	1332	0.8351
S378	1332	0.8375
S336	2276.9	0.8175
S357	137.5	0.8871
S346	917.89	0.8506
S359P1	532.38	0.8911
S347	385.51	0.8607
S352	385.51	0.8607
S356	385.51	0.8607
S358	100.32	0.8480
S357P	285.19	0.7919
S359P2	147.69	0.9065
S359	680.07	0.9343
S338P	683.07	0.9302
S338	683.07	0.9137
S341P	1593.8	0.9390
S341	1593.8	0.9390
S414	582.6	0.9068
S502	582.6	0.9068

S411	1178.2	0.7872
S401	2872.5	0.8272
S415	3.84	0.8428
S402	2876.3	0.9024
S404	2876.3	0.9024
S405	2876.3	0.9024
S407	2876.3	0.9024
S408	3098.2	0.8349
S453	1400	0.8187
S460	1337.7	0.9119
S456	11.357	0.8759
S452	349.07	0.8542
S511	501.44	0.8724
S503	392.1	0.8516
S384P	33.8	0.8686
S52P	535.24	0.1868
S592	244.46	0.8810
S581	147.64	0.8128
S525	31.13	3.2123
S524	504.11	0.8793
S536	7.0011	0.8315
S527	497.11	0.8411
S549	233.72	0.8263
S550	247.55	0.8582
S537	15.835	0.7862
S598	13.365	0.9008
S599	2.47	0.9289
S267	308.47	0.9279
S538	310.94	0.8826

Table.S.9. Characteristics of the systems in the experiments

Problem	Streams	Characteristics	Number of streams
P1	2, 3	S	3
P2	2, 3; 5, 6	P	6
P3	2, 11; 9, 12; 4, 5, 6, 7, 8	P	12
	1, 2 (11), 9 (12), 3	R	
P4	Information was not provided		6
P5	4, 5	P	8
	2, 6, 7, 8	M	
P6	2, 3, 4; 2, 3, 5, 6	R	7
P7	5, 9; 8, 10	M	10
	5, 7, 8, 9; 2, 3, 5, 6	R	
P8	2, 3; 6, 7; 8, 9; 10, 11	P	11
P9	1, 2	P	12
	8, 9, 10	R	
	3, 4, 5; 5, 6, 7	M	
P10	1, 2; 8, 9, 10, 13; 5, 11, 12	M	13
P11	2, 3, 6, 7; 2, 7, 9, 8, 5; 9, 12, 10	R	12
	5, 6, 8; 10, 11, 12	M	
P12	5, 6, 7	P	12
	1, 2, 3, 4; 4, 9, 10; 8, 9, 11; 10, 11, 12	M	
P13	12, 13	P	16
	2, 3, 4; 6, 7, 10; 12 (13), 14, 16	R	
	1, 2, 4; 6, 7, 8; 7, 9, 10; 14, 15, 16	M	
P14	1, 2, 3, 4; 21, 22; 23, 24	P	24
	10, 11, 15; 15, 16, 19	M	
P15	10, 12, 22, 18, 7, 5; 26, 8, 5, 1, 3, 15, 26	R	28
	1, 2, 3, 4; 5, 6, 7, 8, 9; 10, 11, 12; 12, 16, 22, 25	M	
P16	319, 316, 378, 312; 456, 460; 592, 581; 549, 550, …	P	50
	408, 453; 441, 401, 402, 404, 405, 407; …	R	
	503, 592, 581; 537, 598, 599; …	M	

*** P: Parallel Streams, R: Recycle, M: Measurement, S: Separation** [1]

[1] EC do Valle, RA Kalid, AR Secchi, A Kiperstok. Collection of benchmark test problems for data reconciliation and gross error detection and identification, Computers and Chemical Engineering. 111 (2018) 134-148.