JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 36, 441-465 (2020)
DOI: 10.6688/JISE.202003_36(2).0018

A Parametric and Non-Parametric Approach for High-Accurate Outlier Detection

MOHAMED JAWARD BAH AND HONGZHI WANG
Massive Data Computing Laboratory
School of Computer Science and Technology
Harbin Institute of Technology
Harbin, 150001 P.R. China
E-mail: {easybah; wangzh}@hit.edu.cn

Outlier detection is an essential problem that has been studied in a wide range of applications in diverse fields. One common approach to outlier detection is using statistical models, but these methods have inherent challenges and drawbacks. For instance, in providing optimal solutions that will enable the idea of detecting outliers more effectively with a high detection rate and in minimizing the computational cost. Many statistical techniques that have been proposed are classified into mainly parametric and non-parametric methods, and to the best of our knowledge, evaluating and deciphering the effects of these methods against each other remains to be an open research direction, and most of these statistical methods proposed earlier have not shown high outlier detection accuracy. In this paper, under the umbrella and generalization of statistical approach, we propose Gaussian Mixture Model for Outlier Detection (GMMOD) for the parametric approach and Kernel Density Estimation for Outlier Detection (KDEOD) algorithms for the non-parametric approach, for solving the problem of detecting outliers more effectively and in improving the outlier detection accuracy. The proposed methods are applied to real-world datasets, and our experimental results show that even though both techniques perform well, KDEOD shows favorable by a smaller margin in most cases when compared to GMMOD and both show improved performance over their similar comparative algorithms.

Keywords: outlier detection, parametric, non-parametric, Gaussian mixture model, kernel density estimate

1. INTRODUCTION

In recent times, detecting outliers remains to be a significant and all-embracing research branch in data mining due to its extensive use in a broad range of applications such as in RFID [14], sensor networks [1], trajectories [12] and health diagnosis in medicine [6]. Despite the ambiguity in giving a clear definition of outliers, it is predominantly considered as a data point, which is notably different from others or does not heed to the expected normal behavior [2].

Over the years, several techniques have been proposed to detect outliers. In our prior review work [41], we classified outlier detection techniques into various categories, namely: statistical-, distance-, density-, clustering-, ensemble-, and learning-based techniques. In this paper, we focus mainly on statistical-based techniques which are classified into parametric and non-parametric methods. They are generally faced with inherent

Received October 2, 2018; revised June 8, 2019; accepted July 25, 2019.
Communicated by Wang-Chien Lee.
* This paper was partially supported by NSFC Grant U1509216, U1866602,61602129 and Microsoft Research Asia.

challenges and setbacks such as reliability of the quality of results produced, adaptation in multidimensional scenarios, challenges for real-time applications, the high computational cost in dealing with multivariate data and the scalability with respect to increaseing data dimensionality.

At the onset of outlier detection research, outlier detection was predominantly based on assumptions regarding the statistical distribution. Such assumptions are difficult to embrace because of the large and complex state of current data. The challenges of designing an effective statistical model and the evaluation of these models continues to be a significant problem since large datasets or databases demand practical, efficient, scalable and flexible tests for the process of detecting outliers without fully modelling the whole data initially. In addition to designing suitable models for the data, it is also significant that these models can correctly detect outliers with higher detection accuracy, which has been the shortcomings of some previous methods [15, 20, 34]. For our base-line algorithms [15, 20], even though they are designed for outlier detection problems whose most significant evaluation metric is the improved detection accuracy, still fail to give optimal detection accuracy.

With the above challenges and problems, it is thus clear that the importance of accurately detecting outliers is a significant step in data analytics as this can help discover vital knowledge and make better decisions about the data. An improved detection accuracy gives data with fewer outliers, which are significant for building reliable systems and in improving the output performance of applications making use of data. Therefore, among the challenges faced with this technique, we focus on addressing two key concerns – (1) to enable the idea of detecting outliers more effectively with improved detection accuracy, and (2) to design a suitable outlier detection model to fit real-world dataset.

To achieve the above goals, since the foundational concept of our paper is to evaluate the effect of parametric and non-parametric methods against each other, we first propose two methods for each case to extend our focus and then solve the above challenges mentioned earlier and evaluate the approaches against each other with some baseline algorithms. For the parametric instance, we choose the Gaussian mixture model which has been used in a wide range of application areas [16, 21, 26] including the detection of outliers [15, 34, 37]. However, using this model to solve the outlier detection problem remains to be a cumbersome one because it has inherent disabilities. For instance, it is faced with high computational complexity, performance issues for increased dimensional data, and in correctly detecting outliers with improved accuracy. Using these drawbacks as our motivation, we propose a Gaussian Mixture Model for Outlier Detection (GMMOD) as a suitable model to fit the dataset and to detecting outliers effectively with improved detection accuracy. To achieve the above goal, for a given observed dataset, we apply the expectation maximization algorithm that helps to learn the parameters of the distribution and optimize the model parameters to maximize the likelihood. This is then used to fit in our Gaussian mixture model. We use the posterior probabilities, which are the responsibility of each cluster for a single observation and then calculate the probability density function for a single representation in each cluster. The points that are flagged as outliers are considered as those with low likelihood under a chosen prior, and these are detected by investigating each data point's probability.

While for the non-parametric method, we choose kernel density estimate model which has been used to solve various issues for instance in improving the quality of density-based outliers, since the grid-based density estimation does not show optimal performance [20]. Another area is in solving the problem of the curse of dimensionality when the samples are large. Over the years, the primary focus of most literature is using this technique to enhance higher detection accuracy for outlier detection problems [7, 9, 20]. Using similar motivation, we propose Kernel Density Estimation for Outlier detection (KDEOD) as a suitable model to fit the dataset and to improve on the outlier detection quality and accuracy. We adopt the optimal bandwidth selected and use the Gaussian kernel function as our kernel type among the many others. Also, we use the kernel estimate to perform the density estimation to estimate the ground-truth density. The points that are identified as outliers are points that are not created by the bulk of the data points.

The proposed methods are heavily based on existing techniques mentioned earlier; however, to the best of our knowledge, this paper is different from others. We are the first to propose a comparative statistical-based approach that experimentally evaluates and compares the performance of parametric and non-parametric methods against each other on several real-world datasets. Also, we simultaneously address the challenges of fitting the data to a suitable model and in effectively detecting outliers with improved detection accuracy. The two proposed methods will contribute to the goal of outlier detection correctly and efficiently. They will also serve as significant contributions to existing works that try to design statistical-outlier detection models to fit current large and complex data with improved outlier detection accuracy. It will further serve as an evaluation in understanding the comparative performance of two popular statistical-based classification methods. The paper will undoubtedly give researchers an accurate understanding of the effects of both approaches to outlier detection problems and aid them in understanding the areas that these techniques are most feasible.

The remaining section of this article is organized as follows: In Section 2, we introduce the key definitions of parametric and non-parametric methods and then give the related works as a build-up to our study. We then introduce in Sections 3 and 4 our two proposed approaches, the GMMOD and KDEOD. In Section 5, we give the performance evaluation of the two methods through experimental analysis using real-world dataset and further discuss the results. Finally, the conclusion is given in Section 6.

2. RELATED WORKS

Detecting outliers using statistical techniques can be done in supervised, semi-supervised, and unsupervised style [4, 11]. Several methods have been proposed using wide varieties of models [3, 8, 33, 41], but we limit our discussion to those related to our work. The Local Outlier Factor (LOF) [22], which is one of the baseline algorithms makes use of the *k*-nearest neighbor. In the KNN set of each point, LOF makes use of local reachability density (lrd) and compares it with those of the neighbors of each participant of that KNN set. LOF requires the computation for all objects in the dataset, which is expensive and might miss possible outliers whose local neighborhood density is very close to that of its neighbors. A technique that is in contrast with LOF was later

proposed by Yang *et al.* [15], which instead of focusing solely on local properties, extend its focus on global features. Here, the authors introduced an unsupervised outlier detection method with globally optimal Exemplar-Based GMM (OEGMM). In their technique, they first realized the global optimal expectation maximization (EM) algorithm to fit the GMM to a given data set. The outlier factor for every data point is considered as the sum of the weighted mixture proportions with the weight signifying the relationship with other data points. In later years, for a more robust approach to outlier detection, the use of GMM with locality preserving projections was proposed by Tang *et al.* [37]. They combined the use of GMM and subspace learning for robust outlier detection in energy disaggregation. This method addresses the research gap in [22] that fails to detect outliers in multiple state processes and multi-Gaussian states.

For the non-parametric literature, Latecki *et al.* [9] using kernel functions proposed an unsupervised approach to outlier detection. The outlier detection process is performed by comparing each point's local density to that of the neighbor's local density. The experimental evaluation of the proposed techniques when compared to [22] results in better detection performance in most cases. However, the method still lacks applicability in very large and high dimensional real-life databases. In [20], one of the baseline algorithms (KDEOS), the authors follow a similar trend by proposing a method that represents a similar pattern to local outlier detection [13]. Just like in previous methods [9, 22], it also performs density estimation and compares them with the local neighborhoods. The only difference lies in applying the standard KDE directly instead of testing non-standard kernels. Later, Gao *et al.* [10] proposed a more robust approach to address some of the previous shortcomings. The method shows improved performance and good scalability for large data sets using kernel-based techniques with less computational time when compared to [22].

3. PARAMETRIC APPROACH

In this section, we introduce the fundamental concepts and algorithm of the proposed parametric approach and then give a comprehensive explanation of how the method works.

3.1 Gaussian Mixture Model for Outlier Detection (GMMOD)

In the parametric case for which there is an explicit assumption for a given underlying distribution model for a specified dataset, we choose the Gaussian mixture model which over the years has shown improved performance in many areas when compared to others [38] in the same category for outlier detection problems. This is because it is easy to implement and is considered as one of the most popular methods used for data analysis since it gives a better explanation of the real data. For learning mixture models, it has proven to be one of the fastest algorithms. In addition to its simplicity, it is also recognized as being able to fit in most datasets, unlike other parametric types. For example, regression techniques are not always suitable to support every kind of data, more especially complex and high dimensional subspace data. Adopting the GMM model by optimizing it shows improved detection accuracy in previous techniques as well as in our experimental study. GMM-based techniques have been used in various application areas

[16, 21, 26], and it is not the first time to be used in the detection of outliers [15, 34, 37].

For instance, Yang *et al.* [15] use this technique to solve the challenges of knowing the number of clusters beforehand and in solving the problems encounter with datasets that are characterized by inconsistent densities. The authors address a similar problem as ours to detect outliers, and they proposed a new method created by using a global optimal modification of EM. Their target is bringing global information to each data point. Unlike ours, which focus on local neighborhood information of each data point, it helps in the correct detection of the outliers. Also, their method of estimating the EM results in convex optimization, whereas our method focused on the simple EM optimization. Tang *et al.* [37] and Reddy *et al.* [34] both proposed novel outlier detection approaches using GMM combining with other techniques. In [34], they used the GMM model to detect outliers in seasonal univariate network traffic and [37] used a hybrid technique of this model with subspace learning (GMM and subspace learning combined) to improve the performance of outlier detection methods in energy disaggregation. For the former and later methods, although their methods showed improved performance as claimed by the authors, however, it is application specific and only designed for univariate network traffic and energy disaggregation, respectively. These techniques lack extensive comparison for other application areas and therefore no guarantee to conclude that they perform effectively well in other scenarios. Our method, on the other hand, is a general-purpose outlier detection method that can be adopted in different scenarios. In our experiments, we tested our method with ten different benchmark datasets, unlike in [37], which only used a single dataset to prove the accuracy of their method. Our work is different from others since it is designed to embrace different real-world dataset. In contrast, other techniques that have been combined with other methods to detect outliers, ours is a straight forward approach. Our technique is purely GMM through the optimization of the EM to detect outliers more effectively and to compare the performance of parametric and non-parametric methods.

Even though existing works have used and modified the model before to fit outlier detection problems, however, adopting this model for outlier detection problems remains to be a challenging one because of its intrinsic disabilities. In addition, these methods whose main objective is to detect outliers correctly, yet fall short in some areas such as with high computational complexity, performance issues for increased dimensional data and in correctly detecting outliers with higher accuracy. Owing to all these challenges, it is imperative to design new techniques that can be able to address these issues. Therefore, similar to the motivation in [37], we also try to improve the outlier detection accuracy except that our method is not limited to only energy disaggregation problems but to a broader range of datasets. We proposed a Gaussian Mixture Model for Outlier Detection (GMMOD) as a suitable model to fit the data and applied it in the process of detecting outliers. In our method, to improve the GMM to effectively detect outliers with improved accuracy for a given observed dataset, we first approach the learning problem by applying the expectation maximization algorithm that helps to learn the parameters of the distribution and optimize the model parameters to maximize the likelihood. This learning technique dramatically improves detection accuracy while reducing the report of false positive cases. This is then used to fit our Gaussian mixture model. The points that are flagged as outliers are those considered with low likelihood under a chosen prior, and these are detected by investigating each data point's probability. Calculating each data

point's probability will result in increased computational demand. However, it ensures that the outliers are detected correctly with improved accuracy, which is the overall objective of the paper.

The Gaussian mixture models are unsupervised probabilistic learning methodology for representing normal distribution in data. They are known to be models that supposedly explain that data points are produced from a mixture of a definite number of Gaussian distributions that has unknown parameters [20]. They are usually represented as the weighted sum of M component Gaussian densities.

Mathematically, for a univariate case, suppose we have an observation $X = \{x^{(1)}, \ldots, x^{(N)}\}$ for a mixture of M components with parameter ϑ, we represent the one-dimensional model equation as:

$$p(x \mid \vartheta) = \sum_{i=1}^{M} \omega_i N\left(x \mid \vartheta_i\right). \tag{1}$$

Where $N = (x|\vartheta_i)$ is the distribution and ω_i is consider as the prior probability of choosing each ith component. It serves as the positive mixing component that satisfies this condition:

$$\sum_{i=1}^{M} \omega_i = 1. \tag{2}$$

With the assumption that each observation is from M components – the component of the mixture is represented as the Gaussian density $\omega_i > 0$ and $\vartheta_i = (\mu_i, \sigma_i)$, with μ_i as the mean and σ_i as the variance. $\vartheta = (\omega_1, \ldots, \omega_m, \vartheta_1, \ldots, \vartheta_m)$ is the collection of the model parameters, and the distribution $N = (x| \vartheta)$ is represented as:

$$N(x \mid \vartheta) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2} \right). \tag{3}$$

In our case, we adopt the multivariate approach. For a mixture of M components with mean μ_i and covariance matrix σ_i, we represent the multi-dimensional model equation as:

$$p(x) = \sum_{i=1}^{M} \omega_i N\left(x \mid \mu_i, \Sigma_i\right) \tag{4}$$

where

$$N(x \mid \mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^M \mid \Sigma_i \mid}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i) \right). \tag{5}$$

We use the posterior probabilities, which are the responsibility of each cluster for a single observation and then calculate the probability density function for a single observation in each cluster. This is explained in details in our subsequent section. Our approach does not only estimate the σ of the Gaussian kernel but the set of the mixture component parameters. The points that are flagged as outliers are considered as those with low likelihood under a chosen prior, and these are detected by investigating each data point's probability.

3.2 Learning Model and Algorithms

In our proposed approach, given each Gaussian density with its mean μ and covariance $\sum$, referred to as the components of the mixture M, we use the Expectation Maximization (*EM*) which is the most prevalent used method for estimating the mixture model's parameters. The *EM* with respect to GMMOD can be defined as:

Definition 1: Expectation Maximization, *EM*

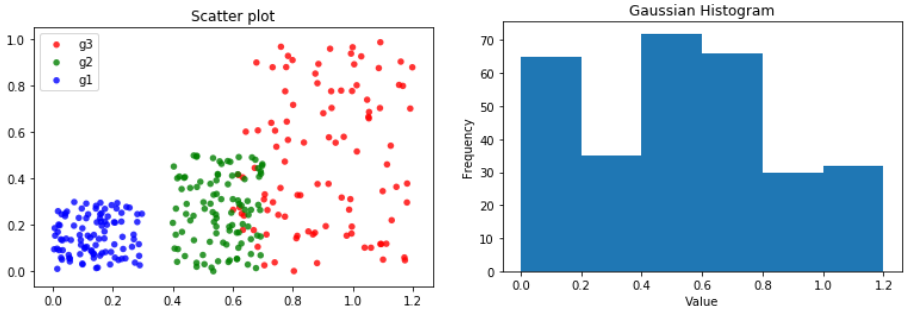
EM is a kind of numerical method that estimates the parameters of a mixture given a fixed amount of mixture components, and it also computes the maximum likelihood estimate of parameters of parametric kind of mixture distribution models [20]. It is an iterative algorithm that comprises two steps. The expectation step (*E*-step) and the maximization step (M-step). In Example 1, we give a clear step by step procedure of how we applied *EM* in GMMOD.

Example 1: Let consider Fig. 1 (a) with three Gaussians to compute the mean and variance of each distribution. Such an approach can fairly give a good estimate as can be seen from the histogram in Fig. 1 (b). From our intuition, we estimate that the mean of g1, g2, and g3 from the distribution are 0.15, 0.55, and 1.0, respectively. However, these estimates might not be accurate, and the method of estimation can be improved with a better estimate of the mean and variance. In order to do this, we need to apply the EM algorithm to generate the best hypothesis for each distribution. The best hypothesis, in this case, is the maximum likelihood that maximizes the probability that a particular data point stems from M distribution with mean μ_m, and variance σ_m^2. The estimated mean and variance are defined respectively as in Eqs. (6) and (7).

$$\mu_m = \frac{\sum x_i}{N} \tag{6}$$

$$\sigma_m^2 = \frac{\sum (x - \mu_i)^2}{N} \tag{7}$$

Step 1: To start, we randomly assign the estimated values from the histogram plot in Fig. 1 (a), to estimate the initial value of each mean μ_m and variance σ_m^2.



(a) (b)

Fig. 1. (a) A scatter plot; and (b) A histogram drawn from three normal distributions.

Step 2: We then consider each data point from the distribution and then check the likelihood that the data point was generated from the mean μ_m and variance σ_m^2 for each set of the distribution parameter. Since we have three parameters, we create a response thrice. That is, the probability that $x^{(i)}$, …, $x^{(N)}$, were drawn from $N(\mu_1, \sigma_1^2)$, $N(\mu_1, \sigma_2^2)$, and $N(\mu_1, \sigma_3^2)$. We then substitute i = 1, 2, 3 in Eq. (2), to obtain the normal density function for each. The probability of x_i belonging to $N(\mu_1, \sigma_i^2)$,

$$p(N(\mu_i \mid \sigma_i^2) \mid x_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2} \right). \tag{8}$$

Step 3: Since each of the probabilities from Step 2 gives us only the partial result, we then consider the likelihood of choosing $N(\mu_1, \sigma_1^2)$, $N(\mu_2, \sigma_2^2)$, and $N(\mu_3, \sigma_3^2)$. This step helps us to check the likelihood of a data point, for instance, x_i belonging to a particular class C_m. This is referred to as the responsibilities of each distribution for every data point, and we represent it as:

$$p(C_m \mid x_i) = \frac{p(C_i)p(x_i \mid C_m)}{\sum_{j=1}^{m} p(C_j)p(x_i \mid C_j)}. \tag{9}$$

Step 4: We then repeat this probabilistic assignment for each data point until it gives the initial data into M number of clusters, which will then let us update the initial estimate hypothesis to the subsequent hypothesis.

Step 5: The estimation step and updating step are repeated until they converge to the final hypothesis value.

Although the major trouble in learning Gaussian mixture models is with unlabeled data in which we are often not aware of which data points is from which hidden component. However, in our case, adopting the EM algorithms serves as an advantage in addressing this drawback. The EM algorithms ensure that even though the convergence towards the peak takes time, however, for each iteration, the likelihood is improved. The algorithm does not demand more effort to be implemented, and the parameter constraints are addressed indirectly. When we compare the improved EM with gradient methods, which is a direct maximization of the likelihood functions, we notice that such methods entail more analytical preliminary work to get the gradient of the likelihood function. Also, challenges like memory constrain, instabilities, and converge issues will all arise, especially when the estimated number of parameters is high. This result in numerical difficulties. Owing to these disadvantages, we can see that EM has more advantages, and thus, we adopt it in our method to enhance the outlier detection accuracy to produce better results.

3.3 The GMMOD Algorithm

Algorithm 1 first describes how the proposed GMMOD algorithm works in our experiment. From the selected dataset, we split the data into 80% for training and 20%

for testing. We selected the features that best describe the data and then normalized the values. To apply the algorithm to fit the model, we first selected the number of components, which was chosen carefully to create a useful model. This is because we want to avoid overfitting and not to lose the generalization of the distribution model's behavior and also to prevent the use of few Gaussians which will result in losing flexibility in describing the normal state. Usually, too few components signify inaccurate modeling of the data, *i.e.* underfitting the data whereas, too many components result in an over-fit model with singular covariance matrices.

We then applied the EM technique in Algorithm 2, which helps us to estimate the model's parameters and to know the number of components that are better suited for the GMMOD fit. We applied the E-Step to calculate the expectation of the task of the component given the model parameters for each data point, while the M-Step entails updating the values got from our E-step; that is the maximization of the expectation in the E-Step. In synopsis, the two key steps in the EM Algorithm are as follows:

E-step: During this step, we calculated the expected value of the cluster assignments based on the hypothesis of the distribution class parameters. For each point, we found the weights by encoding the probability of the membership in each cluster. For all values of i we have a case in which β_{im} is considered to be the probability that x_i is generated by C_m. Therefore, $\widehat{\beta}_{im} = p(C_m \mid x_i, \widehat{\omega}, \widehat{\mu}, \widehat{\Sigma})$

$$\widehat{\beta}_{im} = \frac{\widehat{\omega}_m N(x_i \mid \widehat{\mu}_j, \widehat{\Sigma}_j)}{\sum_{J=1}^{M} \widehat{\omega}_j N(x_i \mid \widehat{\mu}_j, \widehat{\Sigma}_j)}. \tag{10}$$

M-step: During this step, we calculated the new maximum likelihood for our hypothesis. For each model's parameter from the E-step, we maximized and updated them. The updated values are calculated as follows[1]:

$$\widehat{\omega}_m = \sum_{i=1}^{N} \frac{\widehat{\beta}_{im}}{N}, \tag{11}$$

$$\widehat{\mu}_m = \frac{\sum_{i=1}^{N} \widehat{\beta}_{im} x_i}{\sum_{i=1}^{N} \widehat{\beta}_{im}}, \tag{12}$$

$$\widehat{\Sigma}_m = \frac{\sum_{i=1}^{N} \widehat{\beta}_{im} (x_i - \widehat{\mu}_m)(x_i - \widehat{\mu}_m)^T}{\sum_{i=1}^{N} \widehat{\beta}_{im}}. \tag{13}$$

For a univariate scenario, the updated variance is

$$\widehat{\sigma}_m = \frac{\sum_{i=1}^{N} \widehat{\beta}_{im} (x_i - \widehat{\mu}_m)^2}{\sum_{i=1}^{N} \widehat{\beta}_{im}}. \tag{14}$$

[1] https://brilliant.org/wiki/gaussian-mixture-model

Algorithm 1: The GMMOD Algorithm

Input:

- $X = \{x^{(1)}, ..., x^{(n)}\}$, $n = 1, \ldots, N$: dataset
- t_r and t_s: training & test data
- μ_M, σ_M, and ω_M: model parameters
- *diag*: covariance type
- K: number of components

Output:

- Process

```
for each dataset X = {x^(1), ..., x^(n)}, n = 1, …, N do
    if elements in X ≠ 0 then
        Read X
        Split X into t_r and t_s
    else
        Preprocess or Clean X
    end if
end for
for each feature in X do
        Normalize data
end for
for each class in X do
    if class count == min then
        Mark outlier
    else
        Mark inlier
    end if
end for
Select/Assign:
    - The number of components. If K is known, use Algorithm 2.
    - Covariance type = diag
    - Number of iterations
Train the model
Fit t_r into the model
Return the generated prediction of t_s
```

Notably, we know that training the parameters of the model heavily relies on maximizing the log-likelihood of the data. In EM, the successive iteration does not in any way affect the likelihood as compared to other gradient maximization methods [25] that have this limitation. Some of the disadvantages of the EM algorithm that we found are that it is a first-order kind, so it converges slowly for a fixed-point solution, and it also tends to ascertain the local maxima and the initial values sensitivity spuriously. If the model that best suits the trade-off between the fit and the number of components is found; they are usually retained. Both Akaike's Information Criterion (AIC) [23] and Bayesian Information Criterion (BIC) [24] can be used to get insight or determine a suitable number of components for the model when the number of components is not specified. From our experiment, we found out that it can be quite tedious to tell whether we need two, three, four, or even more components. In many cases, as the number of components M increases, so does the complexity.

Algorithm 2: The EM Algorithm

Input:

- $X = \{x^{(1)}, ..., x^{(n)}\}$, $n = 1, ..., N$: dataset
- ω_M, μ_M, σ_M, and ε: model parameters & convergence threshold
- M, C_M: number of components and components assignment
- β_{ik}: probability of x_i generated by C_M

Output:

- $\vartheta = (x| \mu_i, \sum_i)$, for $i = 1, ..., M$.

1. At, $t \leftarrow 0$, initialize ϑ
2. **for each** i in samples X **do**
3. Randomly assign and obtain $\hat{\mu}_1, \ldots, \hat{\mu}_M$
4. Set $\forall_i, X, \vartheta$ covariance estimates $\hat{\sum}_1^2, \ldots, \hat{\sum}_M^2$
5. Set component distribution $\hat{\omega}_1, \ldots, \hat{\omega}_M$
6. **end for**
7. ***E*-step**
8. Given ω_M, μ_M, σ_M,
9. **for each** data point $x_i \in X$ **do**
10. Calculate $\forall_i, X$, expectation of assignment C_M as in Eq. (10)
11. **end for**
12. ***M*-step**
13. Maximize *E-step i.e.* update ϑ at time t, ω_M, μ_M, and σ_M, as in Eqs. (11)-(13)
14. Repeat *E-step* and *M-step* until it converges ($\leq \varepsilon$)
15. **if** it converges **then**
16. Get the Maximum Likelihood
17. **else**
18. Repeat until it converges
19. **end if**
20. Return ϑ

In our experiment, upon completing the EM algorithm in Algorithm 2, we fit the model to perform a clustering inference on the GMMOD model. We used Bayes' theorem together with the assumed estimated model parameters to estimate the posterior component assignment probability. So, given the model's parameters for a multivariate case, we inferred from using Baye's theorem that the probability of a particular data point is from the component C_m by

$$p(C_m \mid x_i) = \frac{\omega_m N(x_i \mid \mu_m, \sum_m)}{\sum_{J=1}^{M} \omega_j N(x_i \mid \mu_j, \sum_j)}. \tag{15}$$

Algorithm 1 shows our proposed GMMOD algorithm for a multivariate case for outlier detection and Algorithm 2 shows the EM Algorithm, which is specifically used or needed when the number of components M is unknown a priori. The EM algorithm for GMMOD starts with an initialization step where we assigned reasonable model parameters based on our data. It then iterates over the *E-step* and *M-step* until the parameter estimates converge at some defined threshold for all parameters at iteration time t, $|\vartheta_t - \vartheta_{t-1}| \leq \varepsilon$. The inferring of $p(C_m \mid x_i)$ becomes simpler by knowing ω_M, μ_M, and σ_M.

4. NON-PARAMETRIC APPROACH

In this section, we introduce the fundamental concept and algorithm of the proposed non-parametric approach and then give a comprehensive explanation of how the method works.

4.1 Kernel Density Estimation for Outlier Detection (KDEOD)

For the non-parametric method, we choose the kernel density estimate over histogram [40] as our case study. Besides the histogram, kernel density estimate is probably the most frequently used estimator and is one of the most studied mathematically [39]. KDE, despite its potential for better performance, it mostly comes at the risk of high cost, and it is generally accepted as quadratic complexity [7]. This technique has been used to solve various issues and is not the first proposed method for outlier detection problems as other methods have already been applied using KDE technique [7, 9, 20]. Unlike the others, it is the first non-parametric method we are aware of that simultaneously tries to improve the outlier detection accuracy and used as a case study for comparing non-parametric and parametric approach for outlier detection problems. The proposed technique [9] decline in performance because of the inheritance of LOF's naïve density estimation. They modify the estimates in accordance with LOF deficiency and lose the statistical foundation of the method. Different from their approach, we modified the standard KDE without losing the statistical knowledge and simultaneously maintained it. In our method, to improve on the detection accuracy, we modified the standard KDE by fine-tuning and using an optimal bandwidth selection technique. This is further explained in Section 4.2. We selected the optimal bandwidth for our algorithm and used the Gaussian kernel function as our kernel type among the many others. In [20], KDE has been adopted for instance in improving the quality of detecting density-based outliers in low dimensional datasets and also in solving the problem of the curse of dimensionality when the samples are large. While in [7], the authors focus on improving the computational efficiency by proposing fastKDE method. Their key objective focuses on the computational speed, while we address the effectiveness in accurately detecting outliers even as the expense of its computational speed. We plan to solve the problem of computation cost; as stated for our future studies.

In addition, our technique differs from the others as the existing techniques [7, 9, 20] have not been compared with parametric methods, and those studies do not focus on studying the effect of the dimensionality of the datasets. Since over the years, the primary aim of most literature is to enhance the quality and detection accuracy for outlier detection problems using this technique. With similar motivation, we propose Kernel Density Estimation for Outlier detection (KDEOD) as a suitable model to fit the dataset and to improve on the outlier detection quality and accuracy. KDEOD focuses on local neighbors similar to [9, 22] but a bit different from [9] that only take into consideration the solid statistical foundation. It has the flexibility of comparing the estimate of the local properties as well as taking into accounts those will firm statistical foundation. The points that as outliers are points that are not created by the bulk of the data points. We improved and contributed to the process by selecting the optimal bandwidth and use the Gaussian kernel function as our kernel type among the many others. Also, we use the kernel estimate to perform the density estimation to estimate the ground-truth density.

The points that are identified as outliers are points that are not created by the bulk of the data points

Kernel density estimation is a prevalent non-parametric approach that is used to estimate or visualize the probability density function (pdf) of a continuous random variable, and it is a data smoothing problem [27]. It is non-parametric because it does not assume any underlying distribution for the variable. In our proposed method, to robustly identify outliers in a given data set of sample $x = \{x^{(1)}, ..., x^{(N)}\}$, we chose a suitable bandwidth to provide a more robust density probability estimate. Usually, the probability estimate does not mirror the actual estimate for the data points if the data is contaminated with outliers and noise. The kernel function describes the contribution of sample points to the density values, and it must be non-negative, even and real-valued. Given n data samples, the multivariate KDE [30] drawn from an unknown density $\hat{\rho}$, is generally defined for n samples of dimension D as:

$$\hat{\rho} = \frac{1}{h^D n} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{16}$$

where K is the kernel, n is the sample size, $x = \{x^{(1)}, ..., x^{(N)}\}$, $x = \{x_{id}, ..., x_{id}\}^T$, D is the dimension and h is the bandwidth also referred to as a smoothing parameter, which usually tends to zero when the number of samples(x) tend to ∞. K should satisfy the following conditions

$$K(x) \geq 0, \text{ and } \int_{-\infty}^{+\infty} K(x)dx = 1. \tag{17}$$

Among the different kinds of the kernel (uniform, normal, biweight, triangular, Epanechnikov and tri weight), we use the multivariate normal kernel Gaussian function defined in Eqs. (18) and (19), where the norm vector is denoted as $||x||$. It is important to note that choosing the Gaussian as our kernel function is not the same as fitting the distribution to the Gaussian model because the kernel only represents the weighted function of each point; which is very prudent for the estimation of the unknown distribution. We applied a similar approach as in [9] by using the kernel estimate to determine the ground truth of the density, which is also known as the majority data points of $\hat{\rho}(x)$. However, methods in [9, 20] focus on the local density outlier detection. In our case, the potential outliers will be points that will be un-generated by the data samples $\hat{\rho}(x)$. We performed the density estimation using the Gaussian kernel.

$$K(x) = \frac{1}{(2\pi)^D} \exp\left(\frac{-||x||^2}{2}\right) \tag{18}$$

For n number of points, the kernel density will be

$$\hat{\rho}(x) = \frac{1}{h^D n} (2\pi)^{\left(-\frac{D}{2}\right)} \sum_{i=1}^{n} \exp\left(\frac{-||x - x_i||^2}{2}\right). \tag{19}$$

4.2 Bandwidth Selection

One of the most challenging procedure in producing a good kernel density estimate

is how to use a suitable bandwidth; as it is computationally intensive. A plethora of methods have been proposed to choose the optimal bandwidth [31, 32, 43], but still, a vast majority of these techniques have accepted that there is no ideal approach in selecting the optimal bandwidth. However, there is still a possibility of deciding a reasonable bandwidth for a specific problem at hand. A common way to choose the optimal bandwidth is using the bandwidth that minimizes the optimality criterion (which is a function of the optimal bandwidth). Initially, the Asymptotic Mean Integrated Squared Error (AMISE) was the most commonly used, but its formula can't be used directly since they involve the unknown density function ρ. From many reviews, the consensus that the plug-in selectors [28] and cross-validation selectors [29] are the most useful over a wide range of the dataset. In recent years also, one of the most successful state-of-the-art technique is to estimate the general integrated squared density derivative functions [7].

In our work, we consider the following guidelines to choose the bandwidth. From intuition, a smaller h will result in a lower standard deviation, while conversely, a larger h will result in a larger standard deviation. We, therefore, adopt the first scenario that is, choosing a smaller h when the sample size is large, and the data are more tightly packed. The latter is best used when the sample size is smaller, and the data are sparser. After this process, then a suitably chosen bandwidth through cross-validation is then applied. This step is considered very significant since the importance of selecting an appropriate bandwidth cannot be overemphasized as it regulates the bias-variance trade-off in estimating the density. Over-fitting will mean the bandwidth is too narrow and this will, in turn, lead to a high-variance estimate, whereas under-fitting depicts that the bandwidth is too broad, which will result in a high-bias estimate.

Our selected cross-validation approach amidst sometimes it flaws of under smoothing and breaking down in some cases with large data samples performs better that gives near-optimal results than the others [5, 31] and it thus produces more authentic results for real-world datasets. However, with large dimension d, the computational rate becomes slow. It is very perplexing to decipher high-dimensional distributions from the unset, in most cases, either we result to parametric assumptions that can potentially be inaccurate, or we accept the very slow convergence.

4.3 The KDEOD Algorithm

In this section, we describe how our proposed KDEOD algorithm works, as shown in Algorithm 3. From the selected datasets, we loaded the dataset and split the data into 80% for training and 20% for testing. We selected the features that best describe the data and then normalized the values. To apply the KDEOD algorithm to fit the model, we first selected the kernel type as per the guidelines in our previous section from among other types of kernels to perform the density estimation. We then chose the optimal number of bandwidths as per guideline of Section 4.2 above. We fit the KDEOD function using Eq. (16), to model the data, normalized the score, and then determine the misclassified classes from the data. We wanted to use just the density estimate to deter mine the outliers, but it is not the best of approach as our estimated distribution model happens to be multimodal. We notice there are differences in the densities of data points belonging to different components, and this can be confusing as it might not be an outlier, and some may

have lower densities than the actual outliers. In KDEOD, the shape of the kernel form does not directly influence the approximation that much, but it is important to know that the kernel function parameters thus play an essential role.

Algorithm 3: The KDEOD Algorithm

Input:

- $X = \{x^{(1)}, ..., x^{(n)}\}$, $n = 1, ..., N$: Dataset
- t_r and t_s: training and test data
- K and n: kernel and sample size
- h: bandwidth, $h > 0$

Output:

- Process

```
for each dataset X = {x^(1), ..., x^(n)}, n = 1, ..., N do
    if elements in X ≠ 0, then
        Read X
        Split X into t_r and t_s
    else
        Preprocess or Clean X
    end if
end for
for each feature in X do
        Select and normalized data
end for
for each list of class in X do
    if class count = = min then
        mark outlier
    else
        mark inlier
    end if
end for
Compute & Select the optimal bandwidth, h
Select Gaussian kernel k(x)
At each datum, x_i builds the selected kernel function as in Eq. (16).
Compute and fit t_r into the model
Return the prediction of t_s
```

In synopsis, from Algorithm 3, we can see that executing the kernel density estimate for outlier detection problems is a relative non-trivial task. We first selected the kernel type, used the cross-validation technique; we determined the optimal bandwidth for the KDEOD model, and fit the model to give our outlier predictions. One big difference between the KDEOD and the GMMOD is that the latter makes strong assumptions about the underlying data distribution when these assumptions do not hold these methods deliver inaccurate densities. Moreover, even when their assumptions hold, popular parametric methods can require extensive parameter tuning. In contrast, non-parametric methods such as this KDEOD can model the distance with few assumptions but are in turn much more computationally expensive.

5. EXPERIMENTAL EVALUATION

In this section, we first introduce our experimental setup and then present our data descriptions. Later, we compare the performance among our proposed algorithms with three baseline algorithms LOF [22], OEGMM [15] and KDEOS [20] in terms of well-known evaluation metrics used for outlier detection. In the final section, we discuss our findings.

5.1 Experimental Setup

We conduct an extensive experiment to evaluate our proposed methods and the baseline algorithms on eight real benchmark datasets from an openly provided access of large collection of outlier detection datasets (ODDS)[2]. These datasets can also be downloaded directly from the UCI Machine learning repository[3]. In each of these datasets, we assumed one/set of class(es) as outlier(s) (*i.e.* rare classes) and normal class (inlier class, the most dominant class(es)). We used python to design our source code and ran on Anaconda Navigator platform using Spyder 3.2.4, which is a powerful python IDE. The test platform is on a 64Bit Ubuntu 16.04LTS system with CPU of 3.20GHz X4, with Disk Space of 230GB and Memory 3GB. We used the standard metrics used to evaluate outlier detection problems by calculating the detection rate and the detection accuracy and showing the receiver operating characteristic curve (ROC) and Area Under Curve (AUC) values in a bar chart. For better result, we conducted 15 independent runs of the algorithms in each dataset and calculated the average results.

5.2 Data Description

The approach in this paper is suitable for both synthetic and real-world datasets. In our experiment, we have used ten different real benchmark datasets from an openly provided access to large collection of outlier detection datasets to evaluate the effectiveness of our proposed methods. The datasets are of varying size and have different attributes.

We chose three datasets with dimension 6, four below 15 and three above 30 at random, as shown in Table 1. We decided to categorize them according to the number of dimensions because we want to investigate the effect of dimensionality in outlier detection problems. In some cases, we did some further preprocessing, and in cases where one of the class is already rare, we labeled them as the outlier class. In cases where the dataset contained relatively balanced classes, down-sampling was necessary to create an outlier class. In other instances, multiple large classes were combined to create inliers, and multiple minority classes were combined to create outliers.

In the Vertebral dataset, Abnormal (Ab) and Normal (No) are the two class labels. 'Ab' is the majority class, which is the inlier and 'No' is the minority class. The outlier class was down-sampled from 100 to 30 instances. The Thyroid dataset has three classes not hypothyroid, subnormal functioning and hyperfunction. It has six attributes, with both normal and subnormal classes treated as inliers while the hyperfunction class was considered as the outlier class since it is the minority class. The Mammography has two classes benign and malignant, with a total of 11,183 samples. It also has six attributes, and the minority class are considered as outliers and the remaining as inliers.

---

[2] https://odds.cs.stonybrook.edu
[3] https://archive.ics.uci.edu/ml

Table 1. Dataset description.

Dataset	#Dimensions	# Points	# Outliers
Vertebral	6	240	30
Thyroid	6	3772	93
Mammography	6	11183	260
Shuttle	9	49097	3511
Glass	9	214	9
BreastW	9	683	239
Wine	13	129	10
Satellite	36	6435	2036
Ionosphere	33	351	126
Heart	44	224	10

The Shuttle dataset contains six classes, among which classes 2,3,5,6,7 are included as outliers, whereas class 1 which contain approximately 80% of the data, is the dominant class that forms the inlier class. The Glass dataset have attributes of a multi-class type with several glass types. Here, class 6 which is the minority class is label as the outlier class, while all the other classes make up the inlier class. The Breast Cancer Wisconsin dataset (BreastW) contains two classes, malignant and benign, where the malignant class is marked as the outlier, and the benign class as the inlier. The Wine dataset has three classes with class 2 and 3 marked as inliers and class 1 as outliers.

In the Statlog (Landsat Satellite) dataset, the training and test data are combined. Class 2 is down-sampled to 71 outliers, while all the other classes are combined to form an inlier class. The Ionosphere has a dimensionality of 34, but one is discarded because all the values are zero, so therefore the number of dimensions is 33 with the 'good' class as inliers and the 'bad' class as outliers.

5.3 Performance Evaluation

Usually, to evaluate an outlier detection algorithm, the detection rate and false alarm rate are calculated as shown in Eqs. (20) and (21).

$$DetectionRate(TPR) = \frac{TP}{TP + FN} \tag{20}$$

$$FalseAlarmRate(FPR) = \frac{FP}{FP + TN} \tag{21}$$

Where TP is the true positive, FN is the false negative, FP is the false positive, and TN is the true negative. Obtaining the detection rate tells us how many numbers of outliers are correctly identified while the false alarm rate gives us a relative number of misclassified outliers. Usually, TP and TN are called Sensitivity and Specificity. To show the trade-off between Eqs. (20) and (21), we use the ROC curve, and for further analysis of the quality of the algorithm, we show the AUC values in a bar chart. The AUC is the integral of ROC, which serves as a measure of the detection performance. For an ideal case, the ROC curve has a value of 1 for detection rate while the false alarm is 0. For the AUC, the ideal value is also 1 or 100% as in our case.

5.4 Experimental Results

In this section, Figs. 2-4 show three trained model of three different datasets and the visualization of the inlier and outlier points. It further shows how the predictions are obtained that helps in illustrating the ROC in the ten subfigures of Figs. 5-7 (a)-(c) of the different datasets. The final set of figures, Figs. 8 (a)-(c) show the bar chart plot of the AUC values for the eight different datasets. In Fig. 5 (b), it shows that the KDEOD ROC curve for thyroid for both methods perform better than the GMMOD ROC curve for the thyroid dataset, with thyroid showing better performance than vertebral Fig. 5 (a). In Fig. 5 (c) the KDEOD ROC curve slightly outperformed the GMMOD, with overall, the two proposed methods showed better performance than their competitors for this data. However, the baseline methods also showed good performance with EOGMM having the least performance for this dataset. The curves are considered to be good because it does fit well with the number of dimensions despite the difference in the number of outliers and data points. The AUC score for both methods for the different datasets also follows the same trend as can be seen in the bar chart plot in Fig. 8 (a). The performance difference is minute because the datasets have closely related characteristics, with thyroid data having a higher number of data points than vertebral. KDEOD, which has no underlying assumptions, have enough data points to strengthen its prediction probability amidst the fact that it will be slower than GMMOD because they are inherently characterized by high computational cost.

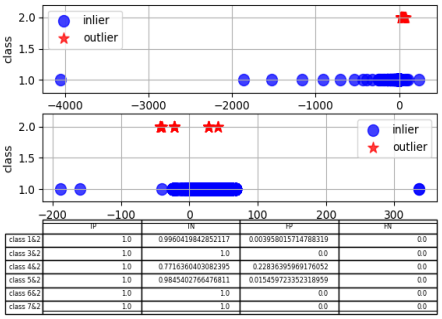


Fig. 2. The shuttle dataset.

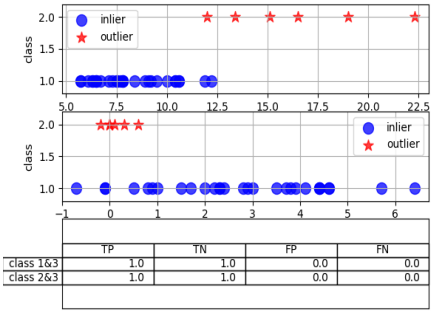


Fig. 3. The thyroid dataset.

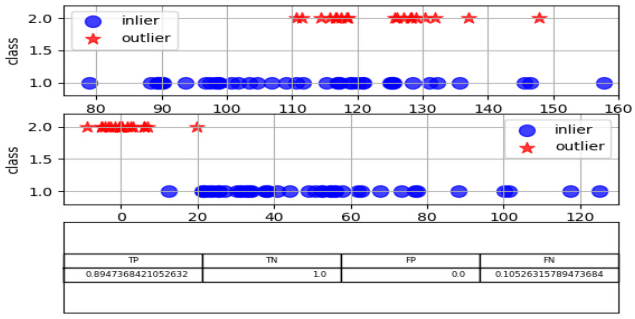


Fig. 4. The vertebral dataset.

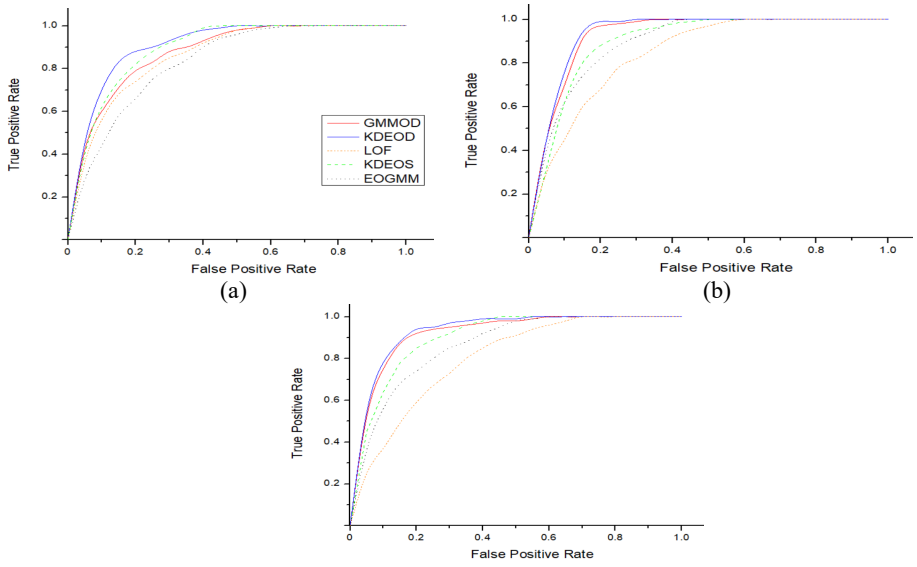


Fig. 5. The (a) Vertebral dataset; (b) Thyroid dataset; (c) Mammography dataset.

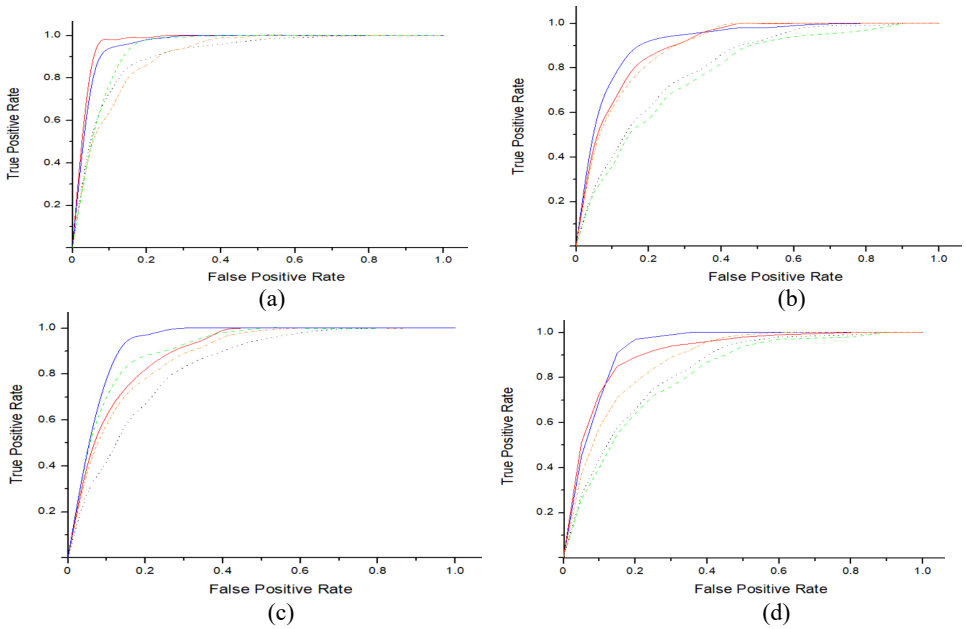


Fig. 6. (a) Shuttle dataset; (b) Glass dataset; (c) BreastW dataset; (d) Wine datasets.

Fig. 6 shows the comparison of the Shuttle, Glass, BreastW, and Wine datasets with the same dimensions except for the Wine dataset. Despite the increase in dimensionality, the shuttle dataset Fig. 6 (a) is close to perfection and performs even better than the previous data for both methods because it has exceedingly number of data points that aid its

learning accuracy. In Fig. 6 (d), the Wine dataset shows similar performance and trend as in Fig. 6 (b) with almost the same number of data points and outliers except for a minute difference in the dimensionality, which does not show much difference in the outlier detection score. The shuttle dataset has higher AUC scores, Fig. 8 (b) for both techniques because the methods can adapt well to deal with the distribution changes. Surprisingly, GMMOD performs better here than KDEOD, possibly owing to the fact of KDEOD's challenge with the curse of dimensionality for higher dimensional datasets. Both methods showed a decrease in performance, on the glass dataset, for instance, among the others, since it represents the most imbalanced data among the four. Fig. 7, finally shows the comparison of the Satellite, Ionosphere, and Heart datasets. In these datasets, the performance dropped when compared to the previous datasets as can be seen from the AUC bar chart in Fig. 8 (c).

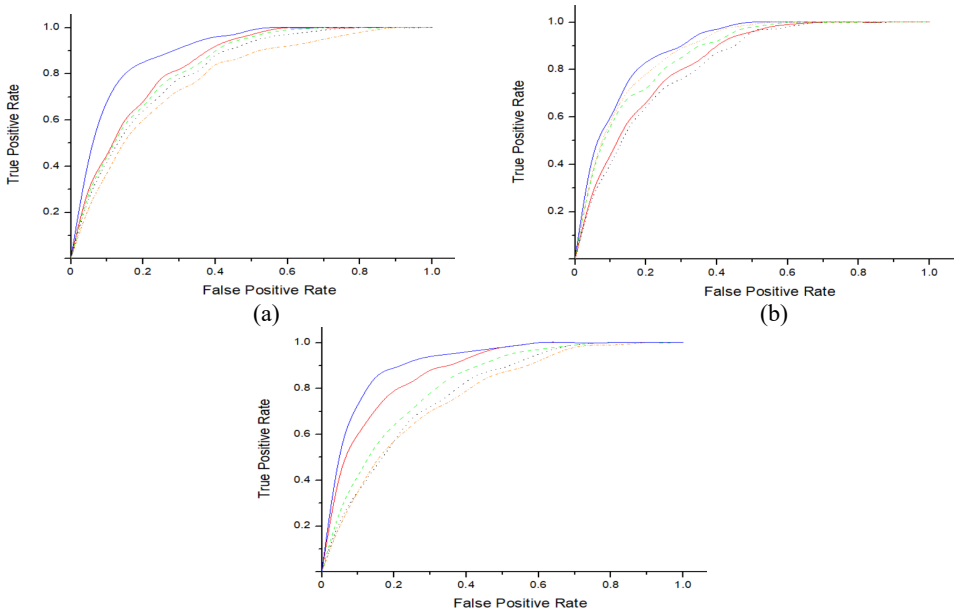


Fig. 7. (a) Satellite dataset; (b) Ionosphere dataset; (c) Heart dataset.

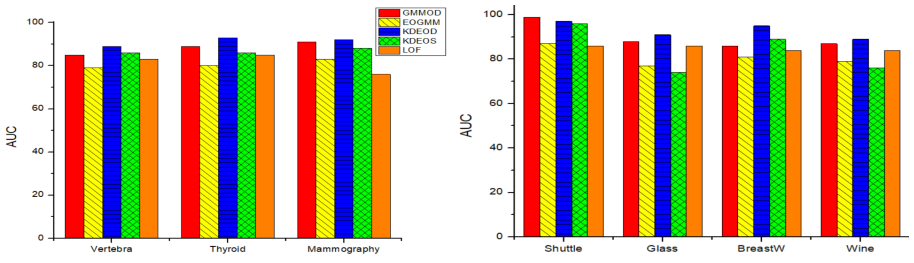
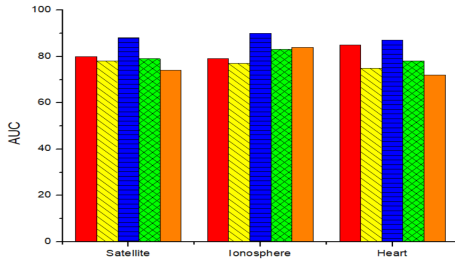


Fig. 8. The AUC values for (a) Vertebral, Thyroid, and Mammography; (b) Shuttle, Glass, BreastW, and Wine; (c) Satellite, Ionosphere and Heart.



(c)

Fig. 8. (Cont'd) The AUC values for (a) Vertebral, Thyroid, and Mammography; (b) Shuttle, Glass, BreastW, and Wine; (c) Satellite, Ionosphere and Heart.

The Ionosphere dataset performed best for both methods as it does have not only the least number of dimensions but also has no class imbalance. The satellite dataset performs better than Heart for both techniques since it has many points, which makes it good for the KDEOD prediction. Another thing we notice here is, some of the baseline methods performance showed almost the same performance as our methods, and the margins are very close. However, throughout these datasets, one cannot clear concluded that one method is superior to the others in all cases. Even though sometimes a method can outperform another method based on the characteristic of the data sets, for example, a smaller number of dataset results in faster learning, however, one cannot clearly conclude based on these factors without taking into consideration other previously mentioned issues.

5.5 Discussion

The two methods show that they are good classifiers and good approach to outlier detection problems since they showed improved performance when compared to baseline methods for our parameter settings and experiments. In some cases, both techniques were able to achieve close to perfection in relatively moderate dimensional data (ROC curve for the Shuttle dataset in Fig. 7 and the trained model in Fig. 2). However, from Figs. 6 and 7, we observe a slight decline for high dimensional data in some cases. The lack of optimal performance is because both GMMOD and KDEOD are not optimal in some high dimensional dataset scenarios and moreover, KDEOD is highly prone to the curse of dimensionality and our choice of cross-validation method for computing the bandwidth also is another major reason. In most cases, both methods made a lower rate of miss-classification in models trained on different classes. KDEOD we believe do outperform GMMOD overall, even though it is sensitive to outliers, but it gives better prediction estimate since it doesn't rely on erstwhile knowledge. GMMOD also performs well, but due to its dependencies and assumptions of the distribution model, the results produce fall short to that of KDEOD with little or no assumptions of the distribution model. It is interesting to note that in some cases, the GMMOD algorithm shows a tendency to reach the maximal true positive rate earlier than KDEOD. Thus, in such case, applications that require a high recall of outliers with the best precision of the outlier performs better. We also observed that there is no direct way to draw conclusion among datasets or methods superiority and performance because it depends on the dataset that is

used, the parameter tuning and how best the algorithm models the data. We cannot clearly conclude that the influence of outliers or dimensionality will predict the performance. However, we believe that if the dataset has a very high dimension, it might face problems of the curse of dimensionality, which consequently will degrade the detection performance. In addition, it will result in slower computational time and high complexity cost.

GMMOD approach, in summary, shows to be easy to use and flexible. However, major constraints are its computational capability, the task of setting the number of mixture models to fit the training dataset. While the KDEOD approach in synopsis shows that it scales well with training samples and is also easy to implement, it is also characterized by some gains like asymptotic consistency, simplification to supplementary density estimators. However, it is also known that when the dimensionality of samples is large, it suffers from the curse of dimensionality. It is also difficult to choose features in kernel density estimation so that the density estimation might be a hard task, and this in turns makes the performance to depend on the choice of the kernel width σ.

6. CONCLUSIONS

Over the years, using statistical techniques, among other techniques to detect outliers has been invaluable in the research domain. In this paper, we presented a parametric and non-parametric approach to outlier detection under the umbrella of statistical techniques. We selected two statistical techniques among the diverse methods as case studies. We proposed GMMOD for parametric methods and KDEOD for non-parametric methods and then compared these methods with other three baselines methods (EOGMM, KDEOS, and LOF). Using real-world datasets, we demonstrated that KDEOD approach is more effective and performs better than GMMOD and overall both further reveals they are successful for outlier detection because they serve the motive behind our research which was to design methods that can effectively fit the different data and give an improved outlier detection accuracy. In most cases, our methods showed good performance when compared to other baseline algorithms based on our experimental study. In our future work, we plan to extend our work using very high dimensional data to investigate the detection performance and remedy previous drawbacks on the demand of computational cost.

REFERENCES

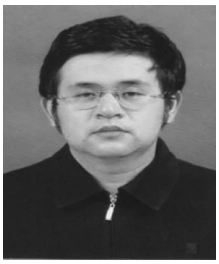
1. A. Ayadi, O. Ghorbel, A. M. Obeid, and M. Abid, "Outlier detection approaches for wireless sensor networks: A survey," *Computer Networks*, Vol. 129, 2017, pp. 319-333.
2. V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, Vol. 41, 2009, pp. 1-58.
3. J. Tamboli and M. Shukla, "A survey of outlier detection algorithms for data streams," in *Proceedings of the 3rd International Conference on Computing for Sustainable Global Development*, 2016, pp. 3535-3540.
4. A. Zimek, R. J. G. B. Campello, and J. Sander, "Ensembles for unsupervised outlier detection: Challenges and research questions [position paper]," *ACM SIGKDD Ex-*

plorations Newsletter, Vol. 15, 2014, pp. 11-22.
5. N. B. Heidenreich, A. Schindler, and S. Sperlich, "Bandwidth selection for kernel density estimation: a review of fully automatic selectors," *AStA Advances in Statistical Analysis*, Vol. 97, 2013, pp. 403-433.
6. G. B. Gebremeskel, C. Yi, Z. He, and D. Haile, "Combined data mining techniques based patient data outlier detection for healthcare safety," *International Journal of Intelligent Computing and Cybernetics*, Vol. 9, 2016, pp. 42-68.
7. T. A. O'Brien, K. Kashinath, N. R. Cavanaugh, W. D. Collins, and J. P. O'Brien, "A fast and objective multidimensional kernel density estimation method: fastKDE," *Computational Statistics & Data Analysis*, Vol. 101, 2016, pp. 148-160.
8. X. Su and C. L. Tsai, "Outlier detection," *Wiley Interdisciplinary Reviews: Data Mining Knowledge Discovery*, Vol. 1, 2011, pp. 261-268.
9. L. J. Latecki, A. Lazarevic, and D. Pokrajac, "Outlier detection with kernel density functions," *Machine Learning and Data Mining in Pattern Recognition*, P. Perner (eds.), LNCS, Vol. 4571, 2007, pp. 61-75.
10. J. Gao, W. Hu, Z. Zhang, X. Zhang, and O. Wu, "RKOF: Robust kernel-based local outlier detection," *Advances in Knowledge Discovery and Data Mining*, J. Z. Huang, L. Cao, J. Srivastava, (eds), LNCS, Vol. 6635, 2011, pp. 270-283.
11. A. Zimek, E. Schubert, and H. P. Kriegel, "A survey on unsupervised outlier detection in high-dimension numeric data," *Wiley Periodicals*, *Statistics Analysis and Data Mining*, Vol. 5, 2012, pp. 363-387.
12. J. Mao, T. Wang, C. Jin, and A. Zhou, "Feature grouping-based outlier detection upon streaming trajectories," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 29, 2017, pp. 2696-2709.
13. E. Schubert, A. Zimek, and H.-P. Kriegel, "Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection," *Data Mining and Knowledge Discovery*, Vol. 28, 2014, pp. 190-237.
14. L. L. Sheng, "Fractal-based outlier detection algorithms over RFID data streams," *International Journal of Online Engineering*, Vol. 12, 2016, pp. 35-41.
15. X. Yang, L. J. Latecki, and D. Pokrajac, "Outlier detection with globally optimal exemplar-based GMM," in *Proceedings of International Conference on Data Mining*, 2009, pp. 145-154.
16. X. Wang, Y. Wang, X. Feng, and M. Zhou, "Adaptive Gaussian mixture models based facial actions tracking," in *Proceedings of International Conference on Computer Science and Software Engineering*, 2008, pp. 923-926.
17. A. Wahab, Q. Chai, K. T. Chin, and K. Takeda, "Driving profile modeling and recognition based on soft computing approach," *IEEE Transactions on Neural Networks*, Vol. 20, 2009, pp. 563-582.
18. S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *Proceedings of the 32nd International Conference on VLDB*, 2006, pp. 187-198.
19. M. Montagnuolo and A. Messina, "Automatic genre classification of tv programmes using Gaussian mixture models and neural networks," in *Proceedings of the 18th International Conference on Database and Expert Systems Applications*, 2007, pp. 99-103.

20. E. Schubert, A. Zimek, and H.-P. Kriegel, "Generalized outlier detection with flexible kernel density estimates," in *Proceedings of the 14th SIAM International Conference of Data Mining*, 2014, pp. 542-550.
21. J. Shen, J. Bu, B. Ju, T. Jiang, H. Wu, and L. Li, "Refining Gaussian mixture model based on enhanced manifold learning," *Neurocomputing*, Vol. 87, 2012, pp. 19-25.
22. M. Breunig, H. Kriegel, R. Ng, and J. Sander, "LOF: Identifying density based local outliers," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, 2000, pp. 93-104.
23. Y. Fang, "Asymptotic equivalence between cross-validations and Akaike information criteria in mixed-effects models," *Journal of Data Science*, Vol. 9, 2011, pp. 15-21.
24. X. Gao and R. J. Carroll, "Data integration with high dimensionality," *Biometrika*, Vol. 104, 2017, pp. 251-272.
25. A. Miller, "Expectation maximization and gradient updates," https://www.andymiller.github.io/2015/10/15/em-gradient.html, 2015.
26. L. Deng and D. Yu, "Chapter-2-gaussian-mixture-models," *Automatic Speech Recognition – A Deep Learning Approach*, Springer, 2014, pp. 6-8.
27. M. Pavlidou and G. Zioutas, "Kernel density outlier detector," *Topics in Nonparametric Statistics*, M. Akritas, S. Lahiri, D. Politis, (eds.), 2014.
28. Z. I. Botev, J. F. Grotowski, and D. P. Kroese, "Kernel density estimation via diffusion," *Annals of Statistics*, Vol. 38, 2010, pp. 2916-2957.
29. A. W. Bowman, "An alternative method of cross-validation for the smoothing of density estimates" *Biometrika*, Vol. 71, 1984, pp. 353-360.
30. D. W. Scott, *Multivariate Density Estimation: Theory*, *Practice*, *and Visualization*, Wiley, NY, 1992.
31. T. Duong and M. L. Hazelton, "Cross-validation bandwidth matrices for multivariate kernel density estimation," *Scandinavian Journal of Statistics*, Vol. 32, 2005, pp. 485-506.
32. S. Chen, "Optimal bandwidth selection for kernel density functionals estimation," *Journal of Probability and Statistics*, Article No. 242683.
33. S. Ranshous *et al.*, "Anomaly detection in dynamic networks: a survey," *Wiley Interdisciplinary Reviews: Computational Statistics*, Vol. 7, 2015, pp. 223-247.
34. A. Reddy *et al.*, "Using Gaussian mixture models to detect outliers in seasonal univariate network traffic," in *Proceedings of IEEE Symposium on Security and Privacy Workshops*, 2017.
35. B. N. Saha, N. Ray, and H. Zhang, "Snake validation: A PCA-based outlier detection method," *IEEE Signal Processing Letters*, Vol. 16, 2009, pp. 549-552.
36. J. Ma, T. Wang, and L. Xu, "A gradient BYY harmony learning rule on Gaussian mixture with automated model selection," *Neurocomputing*, Vol. 56, 2004, pp. 481-487.
37. X. Tang, R. Yuan, and J. Chen, "Outlier detection in energy disaggregation using subspace learning and Gaussian mixture model," *International Journal of Control and Automation*, Vol. 8, 2015, pp. 161-170.
38. C. M. Park and J. Jeon "Regression-based outlier detection of sensor measurements using independent variable synthesis," in *Proceedings of International Conference on Data Science*, 2015, pp. 77-86.

39. M. Pavlidou and G. Zioutas, "Kernel density outlier detector," *Topics in Nonparametric Statistics*, 2014, pp. 241-250.
40. M. Goldstein and A. Dengel, "Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm," *Poster and Demo Track of the 35th German Conference on Artificial Intelligence*, 2012, pp. 59-63.
41. H. Wang, M. J. Bah, and M Hammad, "Progress in outlier detection techniques: A survey," *IEEE Access*, 2019, pp.107964-108000.

Mohamed Jaward Bah received B.Eng. degree in Electrical and Electronics Engineering from the University of Sierra Leone, in 2013, and an M.S. degree in Computer Science from Nanjing University of Information Science and Technology, China 2016. Currently, he is a Ph.D. candidate of Harbin Institute of Technology, China. His research interests include data mining, data stream mining, and big data.



Hongzhi Wang is a Professor and Doctoral Supervisor at Harbin Institute of Technology. He was awarded Microsoft Fellowship, Chinese excellent database engineer and IBM Ph.D. Fellowship. He has published more than 150 papers in refereed journals and conferences. His research interests include big data management, data mining, data quality, and graph data management.