sciendo

VOLUME 19, N° 2 2025
Journal of Automation, Mobile Robotics and Intelligent Systems

# Development Of A Distributed Outlier Detection Method Based On The Alternating Direction Method Of Multipliers

Submitted: 29th February 2024; accepted: 27th March 2024

*Alejandro Cespón Ferriol, Héctor R González Diez, Carlos A. Morell Pérez*

DOI: 10.14313/jamris-2025-009

**Abstract:**
*In data mining, one of the most studied problems is outlier detection, which involves identifying "unusual" data points within a dataset suspected to be generated by a different mechanism than the rest of the dataset. Outlier detection has applications in discovering novel information, detecting bank fraud, identifying system intrusions, and others. However, handling large volumes of data, known as big data, poses a challenge to outlier detection algorithms because the resources of a single computer may not be sufficient to achieve efficient performance. Furthermore, datasets are often stored in distributed environments.*

*The goal of this work is to develop a new distributed outlier detection algorithm based on the solution of the support vector data description using the alternating direction method of multipliers. Mathematical optimization methods and Python language libraries are mainly used for the implementation. As a result, the design and distributed implementation of the proposed algorithm are achieved, which are validated using several test datasets, yielding satisfactory and competitive results compared to existing methods.*

**Keywords:** *outlier detection, big data, method of multipliers*

## 1. Introduction

Outlier detection is one of the most studied problems in data mining, and involves the identification of erroneous or "unusual" data points within a dataset. Outlier detection (OD) finds applications in discovery of novel information, bank frauds, system intrusions [12, 14], as well as in data cleaning for machine learning models that are sensitive to the presence of outliers [28]. Formally, outlier detection can be defined as the problem of finding patterns in the data with unexpected behavior [7]. Outliers are often referred to as anomalies, exceptions, discordant observations, and various other similar terms [3, 21].

Outlier detection is an example of a problem that can be solved using a class of classifiers known as one-class classifiers (OCC), a term proposed by Moya [18]. It solves the problem of finding the boundary that encompasses an entire class, given a dataset that could be contaminated with a small amount of anomalous data.

Tax and Duin [25] describe an OCC called Support Vector Data Description (SVDD). It aims to find the hypersphere with the minimum radius that encloses most data points in a dataset. The data points inside the hypersphere are considered to belong to the target class, while those outside are considered anomalous.

Currently, handling large volumes of data, known as big data, represents a challenge for outlier detection algorithms. The resources of a single computer may not be sufficient to achieve efficient performance for a particular algorithm. Centralized algorithms are becoming less competitive in meeting the time demands of modern applications. Furthermore, due to the increasing sizes of datasets, they are frequently stored in distributed environments [3].

The Alternating Direction Method of Multipliers (ADMM) belongs to the category of distributed optimization and is suitable for solving many machine learning problems [1, 4], especially those that can be formulated or transformed into convex optimization problems [5].

Based on these elements, the main objective of this research is to develop a distributed outlier detection algorithm based on solving the SVDD problem using ADMM. This research aims to contribute to outlier detection in large volumes of data by enabling distributed processing. Moreover, the proposed method does not require centralized data, making it applicable to distributed datasets.

## 2. Concepts and Basic Notation

### 2.1. ADMM

ADMM is a decomposition-coordination technique in which solutions to small local subproblems are coordinated to find the solution to a larger global problem [4]. Depending on the application, it is relatively simple to implement in distributed environments, making it applicable to big data problems [4, 10].

ADMM solves problems of the form:

$$\begin{aligned} &\text{minimize } f(x) + g(z) \\ &\text{subject to: } Ax + Bz = c \end{aligned} \tag{1}$$

Where $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, and $c \in \mathbb{R}^p$. With $f$ and $g$ convex functions. The minimum value of problem (1) would be:

$$p^* = \min\left(f(x) + g(z) \mid Ax + Bz = c\right)$$

Where *min* refers to the minimum. The (augmented) Lagrangian for this problem would be:

$$L_p(x,z,y) = f(x) + g(z) + y^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2$$

And the solution by ADMM consists of the iterations:

$$x^{k+1} := \underset{x}{\operatorname{argmin}}\ L_p(x, z^k, y^k) \tag{2}$$

$$z^{k+1} := \underset{z}{\operatorname{argmin}}\ L_p(x^{k+1}, z, y^k) \tag{3}$$

$$y^{k+1} := y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \tag{4}$$

with $\rho > 0$. As can be seen, the variables $x$ and $z$ are updated alternately, hence the name of the algorithm. The variable $y$ is called the dual variable.

ADMM can be written in a different, often more convenient form, known as the scaled form:

$$x^{k+1} := \underset{x}{\operatorname{argmin}}\ (f(x) + \frac{\rho}{2}\|Ax + Bz^k - c + u^k\|_2^2) \tag{5}$$

$$z^{k+1} := \underset{z}{\operatorname{argmin}}\ (g(z) + \frac{\rho}{2}\|Ax^{k+1} + Bz - c + u^k\|_2^2) \tag{6}$$

$$u^{k+1} := u^k + Ax^{k+1} + Bz^{k+1} - c \tag{7}$$

where $u = \frac{1}{\rho}y$. The dual variable is scaled.

Consensus problems are part of the field of distributed optimization and have historically been related to the ADMM. The consensus optimization problem consists of a single global variable with the objective and constraint terms divided into N parts:

$$\text{minimize} \sum_{i=1}^{N} f_i(x_i) \tag{8}$$
$$\text{subject to: } x_i - z = 0 \quad i = 1...N$$

This is called the global consensus problem.

A common variation of the problem (8) is the introduction of a regularization term in the objective function $g$, which represents a regularization term:

$$\text{minimize} \sum_{i=1}^{N} f_i(x_i) + g(z) \tag{9}$$
$$\text{subject to: } x_i - z = 0 \quad i = 1...N$$

The solution by ADMM to problem (9) would be (in the scaled form):

$$x_i^{k+1} := \underset{x_i}{\operatorname{argmin}}\ (f_i(x_i) + \frac{\rho}{2}\|x_i - z^k + u_i^k\|_2^2) \tag{10}$$

$$z^{k+1} := \underset{z}{\operatorname{argmin}}\ (g(z) + \frac{N\rho}{2}\|z - \bar{x}^{k+1} - \bar{u}^k\|_2^2) \tag{11}$$

$$u_i^{k+1} := u_i^k + x_i^{k+1} - z^{k+1} \tag{12}$$

where $x_i$ are the local primal variables at each node, $u_i$ are the local dual variables, $z$ is the consensus (global) variable, and $\bar{x}$ and $\bar{u}$ are the averages of the primal and dual variables, respectively.

For ADMM with consensus, the residuals are in the form of vectors:

$$r^k = (x_1^k - z^k, ..., x_N^k - z^k),$$
$$s^k = -\rho(z^k - z^{k-1}, ..., z^k - z^{k-1})$$

whose quadratic norms are:

$$\|r^k\|_2^2 = \sum_{i=1}^{N}\|x_i^k - z^k\|_2^2, \quad \|s^k\|_2^2 = N\rho^2\|z^k - z^{k-1}\|_2^2$$

### 2.2. SVDD

SVDD, proposed in [25], is an unsupervised learning method that is very useful for data description and anomaly detection. To describe the dataset, this model finds a hypersphere that encloses most of the data, minimizing the possibility of accepting anomalous data within it.

SVDD is formulated as an optimization problem in the following way: Given a set of points $x_i \in \mathbb{R}^d$, $i = 1...n$:

$$\underset{R,a,\xi_i}{\text{minimize}}\ R^2 + C\sum_{i=1}^{n}\xi_i \tag{13}$$
$$\text{subject to: } \|x_i - a\|^2 \le R^2 + \xi_i, \quad i = 1...n$$
$$\xi_i \ge 0, \quad i = 1...n$$

Where $a$ is the center of the hypersphere, $R$ is the radius, $C$ is a control parameter, and $\xi_i$ are slack variables. The distance from $x_i$ to the center need not be less than or equal to $R^2$, and these large distances are penalized with margin errors $\xi_i$ for each $x_i$. The parameter $C$ controls the trade-off between the volume of the hypersphere and the errors, i.e., it controls the number of points included in the hypersphere.

Another approach to the data description problem can be found in [23,24]. It is based on finding a hyperplane that separates the dataset from the origin with the maximum possible margin. The formulation would be:

$$\underset{w,\rho,\xi_i}{\text{minimize}}\ \frac{1}{2}\|w\|^2 - \rho + \frac{1}{vN}\sum_{i=1}^{N}\xi_i \tag{14}$$
$$\text{subject to: } w \cdot x_i \ge \rho - \xi_i \quad \forall i \quad \xi_i \ge 0$$

Where $w$ is the weight vector of the hyperplane, $\rho$ is the separation margin, $x_i$ are the instances of the dataset, $N$ is the number of instances, $\xi_i$ are slack variables to account for possible errors, and $v \in (0,1)$ is a regularization parameter that indicates the fraction of data points that should be separated (equivalent to the $C$ parameter). Because of this parameter, this method is known as $v$-SVC.

Although the hyperplane is not a closed boundary, it provides solutions comparable to the original SVDD problem when the data is preprocessed to unit norm [25].

## 3. Related Work

### 3.1. Convex Optimization in OCC and Outlier Detection

Convex analysis methods have found increasing application in outlier detection and related areas. OCC can benefit from the robust algebraic and geometric approximations provided by convex analysis and optimization, allowing efficient solution computation through mathematical optimization [26].

Kernel Mean Matching (KMM) [13] is a method that assists in outlier detection by checking whether the training and test sets follow the same distribution. Its solution is formulated as a quadratic programming problem. A similar task to KMM can be performed with Least Square Importance Fitting (LSIF) [15]. In this case, it is also a quadratic programming problem directly related to the least squares problem.

The convex hull is one of the most commonly used convex analyses approaches in OD. This approach aims to find the smallest convex set that encloses a given set of points. Due to its inherent problem formulation, the convex hull has been used in OCC as a method for outlier detection [2, 6].

In [8], the convex modeling of the SVDD problem and its solution using Lagrange multipliers are analyzed. In addition, [23] presents an OCC based on support vector machines, where the model corresponds to a quadratic programming problem related to SVDD. These approaches are of interest in the present research.

In the field of convex optimization, many works focus on decomposition methods and decentralized algorithms [4,11,17], which naturally lend themselves to be approached from the perspective of distributed optimization algorithms.

### 3.2. Distributed Optimization for Outlier Detection

When working with OD in big data, valuable information can be discovered and it is applicable in various domains. A fundamental challenge of OD in distributed systems is to minimize communication between nodes while maintaining the effectiveness of an algorithm [3]. In [14], an algorithm for OD based on data neighborhood is developed with a distributed and input stream focused approach. The algorithm is based on LOCI (Local Correlation Integral), which allows the detection of contextual outliers, i.e., instances that are considered outliers for a particular subset of the dataset. This work highlights the power of parallel processing for OD with input data streams.

In [22], an implementation for real-time OD using the PySpark variant of Spark for Python is proposed. In this work, the training data is stored in the cloud and a cluster-based solution is presented. Instances outside the formed clusters are considered outliers. The approach also includes a scheduler that periodically re-trains the algorithm, allowing for re-evaluation of decisions previously made on instances. Considering that distributed OD is a relatively unexplored area, [28] proposes Sparx, a new algorithm based on the xStream algorithm.

xStream was originally designed for a single processor, but this proposes a Map-Reduce design using Apache Spark in Python. The environment used does not exchange information between nodes, and the data is decentralized.

In [29, 30], the Python library PyOD is proposed, which provides numerous OD algorithms. The library analyzes various characteristics of these algorithms, including their suitability for multi-core processing. PyOD aims to provide a comprehensive set of OD techniques, making it easier for researchers and practitioners to experiment with and apply different outlier detection algorithms in Python.

## 4. Solving the SVDD Problem with Consensus-Based ADMM

Based on the analysis discussed in 2.1 and 2.2, the following solution to the SVDD problem using consensus-based ADMM is proposed:

$$w_i^{k+1} := \underset{w_i}{\operatorname{argmin}} \sum_{j=1}^{N_i} \left[\rho_i^k - w_i x_{ij}\right]_+ + \frac{\alpha_1}{2}\|w_i - w^k + u_i^k\|^2 \tag{15}$$

$$\rho_i^{k+1} := \underset{\rho_i}{\operatorname{argmin}} \sum_{j=1}^{N_i} \left[\rho_i - w_i^{k+1} x_{ij}\right]_+ + \frac{\alpha_2}{2}\|\rho_i - \rho^k + v_i^k\|^2 \tag{16}$$

$$w^{k+1} := \underset{w}{\operatorname{argmin}} \frac{1}{2\lambda}\|w\|^2 + \frac{J\alpha_1}{2}\|w - \bar{w}^{k+1} - \bar{u}^k\|^2 \tag{17}$$

$$\rho^{k+1} := \underset{\rho}{\operatorname{argmin}} -\frac{\rho}{2\lambda} + \frac{J\alpha_2}{2}\|\rho - \bar{\rho}^{k+1} - \bar{v}^k\|^2 \tag{18}$$

$$u_i^{k+1} := u_i^k + w_i^{k+1} - w^{k+1} \tag{19}$$

$$v_i^{k+1} := v_i^k + \rho_i^{k+1} - \rho^{k+1} \tag{20}$$

In this case, $i = 1..J$, where $J$ is the number of nodes and $N_i$ is the number of instances handled by node $i$. The values of $\alpha_1$ and $\alpha_2$ are assumed to be greater than 0. The $\lambda$ parameter acts as a penalty parameter.

### 4.1. Variable Updates

The minimizations are performed using gradients (or subgradients if necessary) and would be as follows:

$$w_i^{k+1} = w^k - u_i^k + \frac{1}{\alpha_1}\sum_{j=1}^{N} I_{w_i x_{ij} < \rho_i^k}(w_i) x_{ij} \tag{21}$$

where:

$$I_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$

$$\rho_i^{k+1} = \begin{cases} \rho^k - v_i^k, & \rho^k - v_i^k \leq \min(\vec{a}) \\ \rho^k - v_i^k - \frac{N}{\alpha_2}, & \rho^k - v_i^k - \frac{N}{\alpha_2} > \max(\vec{a}) \\ \rho^k - v_i^k - \frac{N_t}{\alpha_2}, & \text{otherwise} \end{cases} \tag{22}$$

$$w^{k+1} = \frac{J\alpha_1}{\frac{1}{\lambda} + J\alpha_1}(\bar{w}^{k+1} + \bar{u}^k) \tag{23}$$

$$\rho = \frac{1}{2\lambda\alpha_2 J} + (\bar{\rho}^{k+1} + \bar{v}^k) \tag{24}$$

$$u_i^{k+1} := u_i^k + w_i^{k+1} - w^{k+1} \tag{25}$$

$$v_i^{k+1} := v_i^k + \rho_i^{k+1} - \rho^{k+1} \tag{26}$$

Based on the proposed updates, the solution algorithm would be as shown in Algorithm 1:

**Algorithm 1:** Distributed SVDD using Consensus-based ADMM

**Input:** X, $\lambda$, J, $\alpha_1$,$\alpha_2$
**Output:** w,$\rho$

/* Initialize variables for $w$ */
$w_i = \vec{0}, w = 0, u = \vec{0}$
/* Initialize variables for $\rho$ */
$\rho_i = \vec{1}, \rho = 1, v = \vec{1}$
**while** *stop* **do**
  /* Update $w_i$ at each node */
  $w_i^{k+1} = \text{argmin}_{w_i} \sum_{j=1}^{N_i} \left[\rho_i^k - w_i x_{ij}\right]_+ + \frac{\alpha_1}{2}\|w_i - w^k + u_i^k\|_2^2$
  /* Update $\rho_i$ at each node */
  $\rho_i^{k+1} = \text{argmin}_{\rho_i} \sum_{j=1}^{N_i} \left[\rho_i - w_i^{k+1} x_{ij}\right]_+ + \frac{\alpha_2}{2}\left(\rho_i - \rho^k + v_i^k\right)^2$
  /* Update consensus variables */
  Update $w$ ; Update $\rho$ ;
  /* Update dual variables */
  $u_i^{k+1} = u_i^k + w_i^{k+1} - w^{k+1}$
  $v_i^{k+1} = v_i^k + \rho_i^{k+1} - \rho^{k+1}$
  /* Update history, objective function, dual and primal residuals, and check termination criteria */
  ;
  Update $\epsilon_{\text{primal}}$, $\epsilon_{\text{dual}}$, $r_{\text{primal}}$, $r_{\text{dual}}$, and objective function;
  **if** *stop* **then**
    break

The variable *stop* models the proposed stopping conditions. The most general condition is the maximum number of iterations, which ensures that the algorithm stops.

It also checks whether the residuals do not exceed a tolerance value that combines absolute and relative variants. Finally, an early stopping criterion is introduced, where the algorithm is stopped if there is no improvement of the best value obtained so far for a given number of iterations.

Given the characteristics of consensus ADMM for the distributed architecture, one node should be used where the results are centralized and aggregated. The flow of the algorithm in each iteration would be to update the parameters $w_i$ and $b_i$ at each node independently using its own dataset and the values of $w$, $b$ and the dual variables from the previous iteration. Once the $w_i$ and $b_i$ values of each node have been obtained, they are centralized and aggregated synchronously to give the $w$ and $b$ of the current iteration. Finally, the dual variables are updated. This workflow was implemented using a map-reduce methodology.

## 5. Results and Discussion

### 5.1. Datasets

To evaluate the performance of the algorithm, we use five datasets: three synthetic, and two real. These datasets vary in the number of instances and dimensions. The synthetic datasets are generated based on probability distributions in the plane, which facilitates the construction of graphs to understand the behavior of the algorithm. The datasets are distributed equitably among the processing nodes.

Table 1 briefly describes the five datasets, the instances row defines the total number of elements in each dataset, while the outliers row defines how many of those instances are anomalous. The three synthetic sets corresponding to Experiments 1, 2 and 3 are constructed in the same way, the instances are (x,y) coordinates where most of the data is uniformly distributed around the points (-2,-2) and (2,2) with radius 1. The rest of the instances are normally distributed around the points (-2,-2) and (2,2) with radius 1. The remaining instances are normally distributed in the rectangle defined from (-4,-4) to (4,4) and are considered anomalous. Experiment 4 is performed on the Glass dataset, which has nine continuous features related to the concentrations of metals in the material. It is initially a classification problem with six classes, but class number 6 is in the minority and instances belonging to this class are considered anomalous.[1] Experiment 5 is carried out with a real dataset related to behavior reports for a physical system: Statlog (shuttle). For the experiments, a version is taken in which the main task is to distinguish the reports that are considered anomalous from those that are not.[2]

**Preprocess** The main step in data preprocessing is the use of the Nystroem approximation [19, 27]. This step not only helps to reduce the dimensionality of the kernel matrix, but also allows for modeling the nonlinearity of the data. The kernel used is the Radial Basis Function (RBF) kernel [20].

**Table 1.** Experimental Results

| Parameters | Experiment 1 | Experiment 2 | Experiment 3 | Experiment 4 | Experiment 5 |
|---|---|---|---|---|---|
| Instances | 420 | 2020 | 300500 | 214 | 46464 |
| Dimension | 2 | 2 | 2 | 9 | 9 |
| Type | Synthetic | Synthetic | Synthetic | Real | Real |
| Outliers | 20 | 20 | 500 | 6 | 878 |
| Nodes | 4 | 8 | 2 | 4 | 4 |
| AUC-ROC OCSVM | **0.95** | **0.92** | 0.95 | **0.76** | **0.83** |
| AUC-ROC SGD-OC | 0.94 | 0.90 | **0.96** | **0.76** | **0.83** |
| AUC-ROC ADMM-OC | **0.95** | 0.91 | 0.95 | **0.76** | 0.81 |

In the experiments conducted, it was also found to be effective to scale the values before applying the kernel. For this purpose, the Robust Scaler from scikit-learn is used, which is more effective than standard scaling because the Robust Scaler is not sensitive to the presence of outliers [19].

Other common preprocessing steps can also apply if necessary, such as data type transformation and missing value handling [16].

### 5.2. Experimental Setup

The same methodology was used for all experiments: A labeled dataset is taken, with one class representing the outliers. The labels are stored and removed from the dataset. The classifier is trained on the unlabeled data, then used to classify the data, and the performance is evaluated using the original stored labels.

This process is performed for three One Class Support Vector Machine classifiers, two of which are available in the scikit-learn library: OCSVM and SGD-OC. The third classifier is the one proposed in this research (called ADMM-oc).

Another important aspect is that for the ADMM-oc and SGD-oc methods the data shown in the AUC-ROC in Table 1 are the result of the average of 10 runs in the first four experiments and 5 runs in the fifth experiment. This process is done because both methods are stochastic.

### 5.3. Metrics

The OD problem can be viewed as a two-class classification problem, where one class represents the outliers and the other class represents the "normal" instances [21]. Even methods that return a score or degree can be reduced to a two-class problem by setting a threshold for the data, above which they are considered anomalous [7]. Therefore, scoring metrics for binary classification problems are valid. An important consideration is the typical class imbalance in such problems, which may cause certain metrics to be unrepresentative of the quality of a given algorithm. Therefore, the AUC-ROC metric is evaluated because it is not sensitive to imbalanced data [9, 12].

### 5.4. Validation

Table 1 presents the results of the experiments, with a description of the datasets and some parameters used for each experiment. It can be observed that

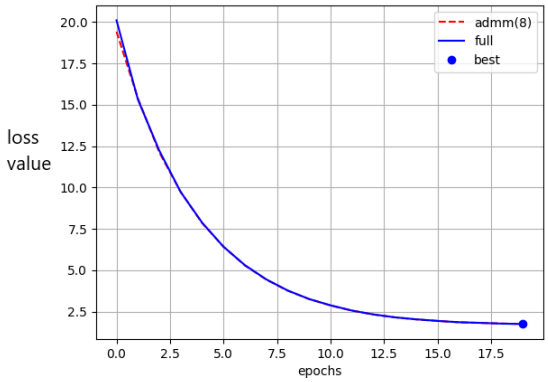


**Figure 1.** Convergence Analysis

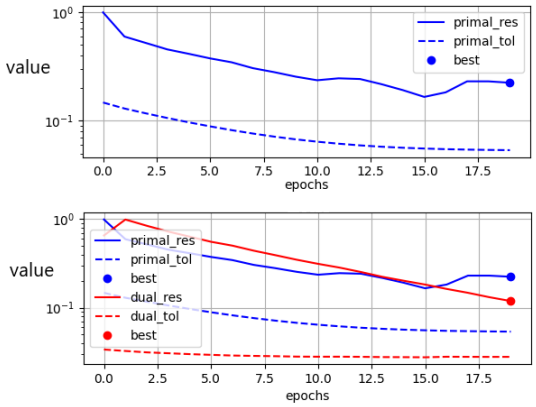


**Figure 2.** Residuals and Tolerances

the AUC-ROC values remain similar to the other two methods compared, regardless of the variation in the datasets.

During Experiment 2, studies were conducted to verify the algorithm's convergence. The results showed that the ADMM-OC algorithm achieves good convergence within a maximum of 20 iterations, as can be seen in Figures 1 and 2.

Considering the two-dimensionality of the dataset used in Experiment 3, a comparison of the contour plots of the classifiers is proposed in Figure 3. As can be observed, the contour plots are almost identical, which supports the similar AUC-ROC values obtained.

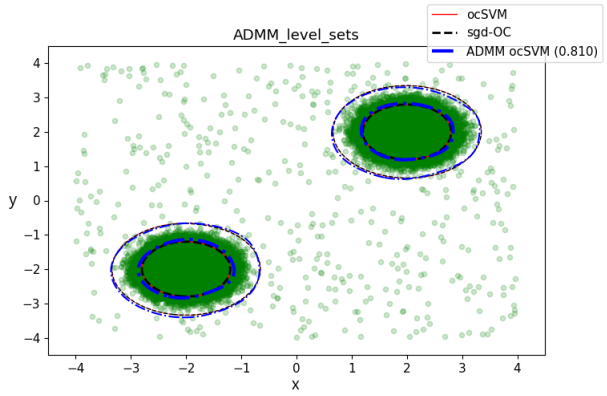


**Figure 3.** Countour Lines

Visually the outermost contour line would represent the boundary between anomalous and normal instances.

Based on the experimental study, it can be concluded that the algorithm maintains good effectiveness compared to other similar methods. The influence of data preprocessing, particularly the Nystroem approximation, on the performance of the algorithm was observed. For two-dimensional datasets, the results were evaluated graphically, visually confirming the effectiveness values obtained. Finally, a detailed analysis of the algorithm's iterations allowed verification of aspects related to convergence and the consensus process.

## 6. Conclusion

Outlier detection is a fundamental area of data mining. It not only contributes to data cleaning, but also to the discovery of new knowledge. However, its application to large volumes of data remains a challenge today. The SVDD problem is useful for outlier detection and modeling it as an optimization problem allows approaching it using iterative methods such as ADMM. The consensus variant of ADMM was used to model the solution of the SVDD problem. The proposed algorithm was implemented. The form of the consensus method allowed this implementation to have a distributed approach. To validate the model, a series of experiments were carried out using synthetic and real datasets with different numbers of instances and dimensions, which verified the competitiveness of the proposed algorithm with respect to other existing ones, obtaining AUC-ROC values in an acceptable range. The graphical analysis of the contour lines in the two-dimensional sets provided clarity to the results obtained.

As future work, it is considered to extend the comparison of the proposed algorithm with other distributed algorithms for OD. It is also proposed to validate the algorithm on non-equal distribution datasets.

## Notes

[1]Original: UCI Repositry: https://archive.ics.uci.edu/static/public/42/glass+identification.zip, Outlier Approach: https://odds.cs.stonybrook.edu/glass-data/.

[2]Original: UCI Repository: https://archive.ics.uci.edu/static/public/148/statlog+shuttle.zip, Version Used: https://dataverse.harvard.edu/api/access/datafile/2711919?format=original.

**AUTHORS**

**Alejandro Cespón Ferriol*** – Universidad Central "Marta Abreu" de las Villas (UCLV), Cuba, e-mail: acferriol@uclv.cu.

**Héctor R González Diez** – Universidad de las Ciencias Informáticas (UCI), Cuba, e-mail: hglez@uci.cu.

**Carlos A. Morell Pérez** – Universidad Central "Marta Abreu" de las Villas (UCLV), Cuba, e-mail: cmorellp@uclv.edu.cu.

*Corresponding author

## References

[1] A. Adler, M. Elad, Y. Hel-Or, and E. Rivlin, "Sparse coding with anomaly detection", *Journal of Signal Processing Systems*, vol. 79, no. 2, 2015, pp. 179–188, doi:10.1007/s11265-014-0913-0.

[2] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls", *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, 1996, pp. 469–483, doi:10.1145/235815.235821.

[3] A. Boukerche, L. Zheng, and O. Alfandi, "Outlier detection", *ACM Computing Surveys*, vol. 53, no. 3, 2021, pp. 1–37, doi:10.1145/3381028.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers", vol. 3, no. 1, 2010, pp. 1–122, doi:10.1561/2200000016.

[5] S. P. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, 2004.

[6] P. Casale, O. Pujol, and P. Radeva, "Approximate convex hulls family for one-class classification", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6713 LNCS, 2011, pp. 106–115, doi:10.1007/978-3-642-21557-5_13.

[7] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection", *ACM Computing Surveys*, vol. 41, no. 3, 2009, pp. 1–58, doi:10.1145/1541880.1541882.

[8] W.-C. Chang, C.-P. Lee, ·. Chih, and J. Lin. "A revisit to support vector data description". Technical report, Department of Computer Science at National Taiwan University, Taipei, Taiwan, 2013.

[9] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, "A comparative evaluation of outlier detection algorithms: Experiments and analyses", *Pattern Recognition*, vol. 74, 2018, pp. 406–421, doi:10.1016/J.PATCOG.2017.09.037.

[10] J. Eckstein and W. Yao. "Understanding the convergence of the alternating direction method of

multipliers: Theoretical and computational perspectives". Technical report, 2015.

[11] M. Fukushima, "Application of the alternating direction method of multipliers to separable convex programming problems", *Computational Optimization and Applications*, vol. 1, no. 1, 1992, pp. 93–111, doi:10.1007/BF00247655.

[12] W. Hilal, S. A. Gadsden, and J. Yawney, "Financial fraud", *Expert Systems with Applications*, vol. 193, 2022, doi:10.1016/J.ESWA.2021.116429.

[13] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting sample selection bias by unlabeled data", *NIPS 2006: Proceedings of the 19th International Conference on Neural Information Processing Systems*, 2006, pp. 601–608, doi:10.7551/mitpress/7503.003.0080.

[14] I. Kalliantzis, A. N. Papadopoulos, A. Gounaris, and K. Tsichlas. "Efficient distributed outlier detection in data streams". Technical report, 2019.

[15] T. Kanamori, S. Hido, and M. Sugiyama, "Efficient direct density ratio estimation for non-stationarity adaptation and outlier detection", *Advances in Neural Information Processing Systems 21 - Proceedings of the 2008 Conference*, 2009, pp. 809–816.

[16] J. D. Kelleher, B. Mac Namee, and D'Arcy Aoife, *Fundamentals of machine learning for predictive data analytics*, MIT Press, 2020.

[17] C.-N. Li, Y.-H. Shao, W. Yin, and M.-Z. Liu, "Robust and sparse linear discriminant analysis via an alternating direction method of multipliers", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 3, 2020, pp. 915–926, doi:10.1109/TNNLS.2019.2910991.

[18] M. M. Moya and D. R. Hush, "Network constraints and multi-objective optimization for one-class classification", *Neural Networks*, vol. 9, no. 3, 1996, pp. 463–474, doi:10.1016/0893-6080(95)00120-4.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: machine learning in Python", *Journal of Machine Learning Research*, vol. 12, no. 85, 2011, pp. 2825–2830.

[20] N. R. Prasad, S. Almanza-Garcia, and T. T. Lu, "Anomaly detection", *Computers, Materials and Continua*, vol. 14, no. 1, 2009, pp. 1–22, doi:10.3970/cmc.2009.014.001.

[21] N. N. R. Ranga Suri, N. Murty M, and G. Athithan, *Outlier detection: Techniques and applications*, Intelligent Systems Reference Library, Springer International Publishing, 2019, doi:10.1007/978-3-030-05127-3.

[22] G. Ranganathan, "Real time anomaly detection techniques using PySpark frame Work", *Journal of Artificial Intelligence and Capsule Networks*, vol. 2, no. 1, 2020, pp. 20–30, doi:10.36548/jaicn.2020.1.003.

[23] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution", *Neural Computation*, vol. 13, no. 7, 2001, pp. 1443–1471, doi:10.1162/089976601750264965.

[24] B. Schölkopf, R. C. Williamson, A. Smola, and J. Shawe-Taylor, "SV estimation of a distribution's support". In: *Neural Information Processing Systems (NIPS)*, 2000, pp. 582–588.

[25] D. M. Tax and R. P. Duin, "Support vector data description", *Machine Learning*, vol. 54, no. 1, 2004, pp. 45–66, doi:10.1023/B:MACH.0000008084.60811.49.

[26] T. Wang, M. Cai, X. Ouyang, Z. Cao, T. Cai, X. Tan, and X. Lu, "Anomaly detection based on convex analysis: A survey", *Frontiers in Physics*, vol. 10, 2022, pp. 873–848, doi:10.3389/FPHY.2022.873848/BIBTEX.

[27] K. Zhang, I. W. Tsang, and J. T. Kwok, "Improved Nyström low-rank approximation and error analysis". In: *Proceedings of the 25th international conference on Machine learning - ICML '08*, New York, New York, USA, 2008, pp. 1232–1239, doi:10.1145/1390156.1390311.

[28] S. Zhang, V. Ursekar, and L. Akoglu, "Sparx: Distributed outlier detection at scale". In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2022, pp. 4530–4540, doi:10.1145/3534678.3539076.

[29] Y. Zhao, *PyOD documentacion release 1.0.9*, USC, 2023.

[30] Y. Zhao, Z. Nasrullah, and Z. Li. "PyOD: A Python toolbox for scalable outlier detection". Technical report, 2019.