Journal of Intelligent Learning Systems and Applications, 2020, 12, 83-108
https://www.scirp.org/journal/jilsa
ISSN Online: 2150-8410
ISSN Print: 2150-8402

Combining Artificial Immune System and Clustering Analysis: A Stock Market Anomaly Detection Model

Liam Close, Rasha Kashef

Electrical, Computer, and Biomedical Engineering Department, Ryerson University, Toronto, Canada
Email: lclose@ryerson.ca, rkashef@ryerson.ca

How to cite this paper: Close, L. and Kashef, R. (2020) Combining Artificial Immune System and Clustering Analysis: A Stock Market Anomaly Detection Model. *Journal of Intelligent Learning Systems and Applications*, **12**, 83-108.
https://doi.org/10.4236/jilsa.2020.124005

Received: August 30, 2020
Accepted: October 11, 2020
Published: October 14, 2020

Abstract

Artificial intelligence research in the stock market sector has been heavily geared towards stock price prediction rather than stock price manipulation. As online trading systems have increased the amount of high volume and real-time data transactions, the stock market has increased vulnerability to attacks. This paper aims to detect these attacks based on normal trade behavior using an Artificial Immune System (AIS) approach combined with one of four clustering algorithms. The AIS approach is inspired by its proven ability to handle time-series data and its ability to detect abnormal behavior while only being trained on regular trade behavior. These two main points are essential as the models need to adapt over time to adjust to normal trade behavior as it evolves, and due to confidentiality and data restrictions, real-world manipulations are not available for training. This paper discovers a competitive alternative to the leading approach and investigates the effects of combining AIS with clustering algorithms; Kernel Density Estimation, Self-Organized Maps, Density-Based Spatial Clustering of Applications with Noise and Spectral clustering. The best performing solution achieves leading performance using common clustering metrics, including Area Under the Curve, False Alarm Rate, False Negative Rate, and Computation Time.

Keywords

Artificial Immune System, Clustering, Anomaly Detection, Financial Data

1. Introduction

The financial stock market is vulnerable to different manipulation attacks to increase or decrease stock value. These manipulation attacks are anomalies in financial trade datasets since they do not follow traditional trading techniques. There are many challenges involved in detecting these anomalies, the first being normal trade behavior becoming anomalous over time or vice versa. This implies that the model must evolve with the financial data over time to model new trading trends effectively. The second challenge of anomaly detection is that some manipulations can result in a minor change as a tactic to go under the radar, emphasizing the importance of the model's generalization. There are also many security restrictions on financial time series trade data, making it hard to obtain the training data. There are few exposed real-world manipulations cases, and most data are partially observable (data with missing information such as the buyer or seller), which can make validation difficult. Most artificial intelligence or machine learning applications in the financial stock market domain aim to predict the value of a stock to execute a buy or a sell [1] [2]. The use of these techniques has not been explored thoroughly in detecting anomalies in financial stock data. A semi-supervised approach was used by [2], which addressed the common issues within this domain, such as parameter tweaking and relying on labeled data. This approach shows promise and suggests a possible avenue of research to expand on.

Current approaches in detecting anomalies in stock market data that use supervised learning [3]-[8] suffer from reliance on a labeled dataset, which cannot be obtained in many scenarios. For example, in [1], the market authority does not disclose the buyers and sellers of each exchange. Besides, there is a limited amount of stock manipulations that get exposed to the public. These difficulties around relying on a labeled dataset encouraged the need for an unsupervised approach. However, current solutions that use unsupervised learning methods [4] [9]-[14] have suffered from problems such as reliance on parameter tweaking, representing time series data effectively, and class imbalance. In [2], a semi-supervised approach is introduced, which mimics the human immune system using the Dendritic Cell Algorithm (DCA). As a result, it does not rely on a labeled dataset and reduces the dimensionality, which decreases the reliance on parameter tweaking, thus providing a more robust solution. In [2], a hybrid combination of DCA and KDE clustering is provided for anomaly detection in financial time series data. The results were compared to the previous leading approaches, and it was determined that the hybrid model outperformed the others in terms of Area Under the Curve (AUC) and False Alarm Rate (FAR). The addition of DCA to KDE increased the AUC by 29% for the Apple dataset and led all other solutions in terms of FAR with the largest rate of 0.68 for the Google dataset. These positive results demonstrate the incentive of investigating the hybrid combination of AIS with other clustering approaches. However, the worst-performing stocks could have better performance if the dataset is expanded.

Previous research studies [15] [16] [17] demonstrate that a DCA inspired approach can effectively detect anomalies and model time series data. In this paper, we adopt the notion of integrating AIS with clustering analysis to better detect anomalies in financial stock market datasets. This paper introduces a hybrid algorithm that combines the DCA from the AIS literature with five well-known clustering approaches, including Kernel Density Estimation (KDE), Density-Based Spatial Clustering of Applications with Noise (DBSCAN), Spectral Clustering (SC), Self-Organizing Maps (SOM), and Cluster-Based Local Outlier Factor (CBLOF) on five stock market time-series dataset. Experimental results on datasets with various configurations and sizes show the effectiveness of the hybrid models in achieving better performance as compared to the individual clustering algorithm as measured by the Area Under the Curve (AUC), False Alarm Rate (FAR), False Negative Rate (FNR), and computational time.

This paper is organized as follows: In Section 2, a background of stock market fraud is given and previous related work for fraud detection. In Section 3, the artificial immune system (AIS) approach and previous work that utilizes the approach is explained. In Section 4, clustering analysis and four-clustering approaches that are implemented in this paper, including KDE, SOM, SC, and DBSCAN, are introduced. In Section 5, the proposed algorithm is presented. In Section 6, the experimental work and results are discussed. Section 7 concludes the paper along with a discussion on future research direction.

2. Stock Market Fraud

Stock market fraud is a common attack with the primary goal of manipulating a stock's price. There are two main forms of manipulations used by attackers called pump and dump and spoof trading. The main goal of pump and dump trading is to increase a stock value and then sell it once it has been increased to obtain the maximum profit possible [1]. The pump and dump manipulations have two main stages; the pumping stage, where the manipulator falsely raises the stock price and the dumping stage. The manipulator sells the stock at the manipulated high price. **Figure 1** shows the pattern of a stock price being manipulated by a pump and dump attack.

In **Figure 1**, the stock's increase is performed by the manipulator entering bid orders for the target stock, which will increase both the price and volume of the stock. This creates a false impression to other investors that the stock is growing in value and leads to others entering bids towards the stock. The next stage is shown in **Figure 2**, where the dumping stage occurs. Once the stock reaches the attacker's target price, they immediately cancel their bid orders and sell the stock at the new overpriced value. As a result, the attacker has made a profit on the stock, and the stock's value returns down to its original price. The second type of manipulation is spoof trading, typically when a large volume trade with a passive bid price is made. The high volume is used to increase the stock price, where the passive price is used to limit suspicion [4]. Similarly, to pump and dump manipulations, the main goal is to artificially drive the stock price up to sell for a higher profit. Figure 2 demonstrates the stock price pattern during a detected spoof trading manipulation by Demonstrate Limited Liability Company in September 2012 [2].

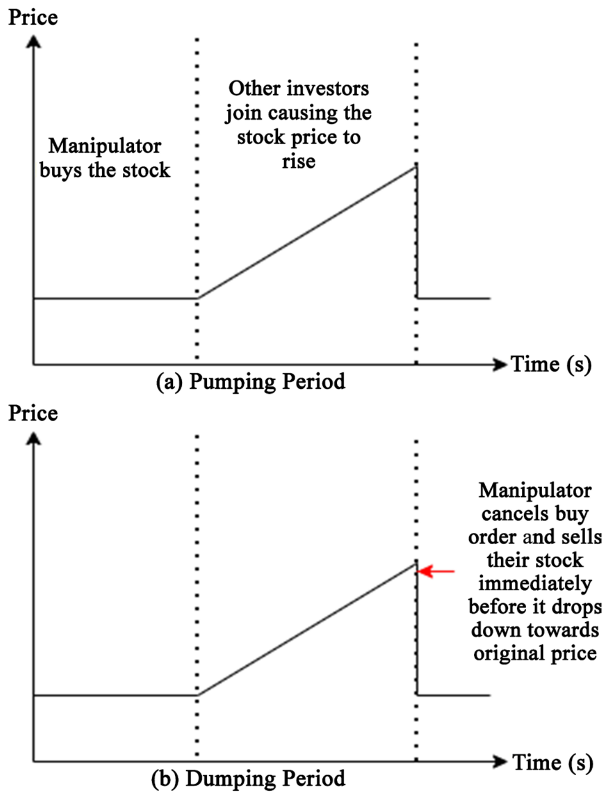
Price
Manipulator buys the stock
Other investors join causing the stock price to rise
Time (s)
(a) Pumping Period
Price
Manipulator cancels buy order and sells their stock immediately before it drops down towards original price
Time (s)
(b) Dumping Period

Figure 1. Pump and dump stock price pattern.

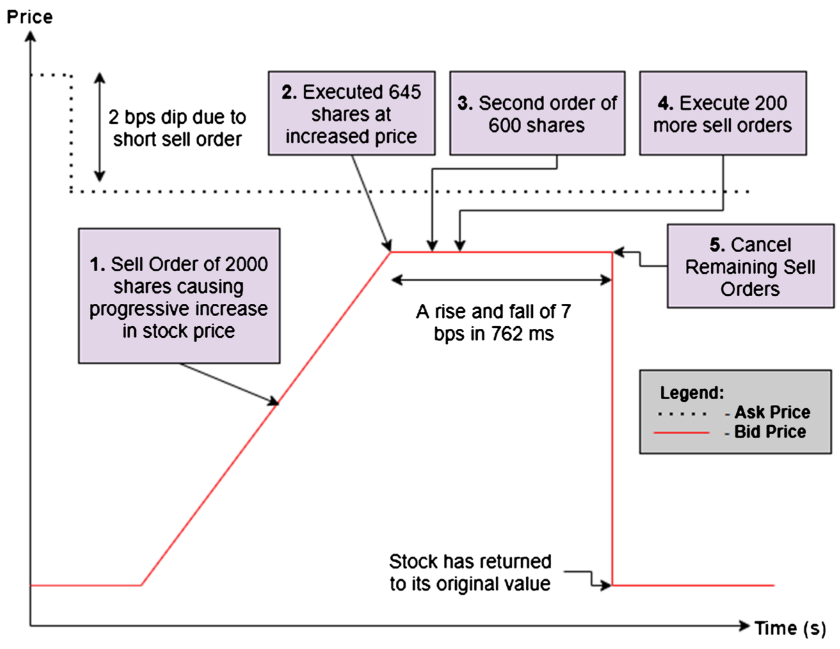
Price
2 bps dip due to short sell order
2. Executed 645 shares at increased price
3. Second order of 600 shares
4. Execute 200 more sell orders
1. Sell Order of 2000 shares causing progressive increase in stock price
5. Cancel Remaining Sell Orders
A rise and fall of 7 bps in 762 ms
Legend:
- Ask Price
- Bid Price
Stock has returned to its original value
Time (s)

Figure 2. Stock price during a spoof trading manipulation attack.

Stock market fraud detection has been a difficult anomaly detection problem, and previous research did not investigate much in stock market prediction models. Supervised learning techniques have been implemented in [3] [4] [7] [8] but suffer from relying on a labeled dataset, which in this domain is considered unrealistic. Some unsupervised approaches have been investigated in [3] [4] [13] [14], with clustering-based methods being the most effective. These approaches are limited by their ability to represent time series data and suffer from class imbalance effectively. Other involved models were introduced in [9] and [10] to improve the limitations of supervised and basic unsupervised approaches. In [9], a Hierarchical Temporal Memory Model was introduced with the main purpose of ensuring that the time series data was represented properly over time. The solution proved to handle time-series data well but is limited by multiple fine-tuning parameters for each dataset it is used on. Authors in [10] have implemented a new approach using Adaptive Hidden Markov Model with the intension of modeling time series data and only needing to be trained on normal trade activity. This solution showed promise in terms of Area Under the Curve but had poor computational complexity and was only tested on a limited dataset. Deep learning approaches to this problem were introduced in [1] [7] to find an innovative solution to detect stock market manipulations. A deep general adversarial network (GAN) approach was first introduced in [1] and it showed a promise in detecting pump and dump manipulations. However, the solution was limited only to detect this one type of manipulation and had a drop-in accuracy when executed on unseen data. A 10-layer deep learning approach called Variational Auto Encoder was implemented in [7], highlighting the ability to detect anomalies without needed a labeled dataset. This solution did not perform as well as some supervised approaches as it has a high False Positive Rate. In [18], they took a different approach to the problem by including textual data from public media content on top of the financial data. Although it did perform well when compared to other solutions, it relied on heavy parameter tweaking and was limited by the difficulty of obtaining the needed textual data. Although there has been previous research in stock market manipulation attacks, the focus on financial data is heavily geared towards stock price prediction.

3. Artificial Immune Systems (AIS)

The natural immune system (NIS) defends the body against its cells' dysfunctions and actions from foreign cells. The authors of [19] explain that a key important feature of the NIS is its ability to react against external, harmful agents while remaining unresponsive to itself. However, NIS can benefit itself by acting against its antibodies. It can eliminate cells that no longer meet an affinity level allowing it to change over time. Artificial Immune Systems (AIS) are computational intelligence techniques based on NIS. AIS is defined in [20] as "Artificial Immune Systems are adaptive systems, inspired by theoretical immunology and observed immune functions, principles, and models, which are applied to problem-solving". They are trained on standard data only and do not get exposed to anomalies as a priori [2]. An AIS system assumes and forms a bias based on this standard activity data and can evolve as the definition of regular activity evolves. An AIS approach is highly applicable to detecting stock market manipulations due to its ability to effectively handle time-series data. It will also avoid any incorrect bias by only training on regular activity, which prevents the negative effects of class imbalance. Another final advantage of this approach is that it does not rely on a labeled dataset that is challenging to obtain in this domain. As AIS proves to be an effective approach when applied to artificial intelligence problems, it has become a respected approach across many domains. AIS is applied to constrained optimization problems in [21], where it proved to achieve competitive performance to the leading approaches. AIS is applied to real-time traffic monitoring systems in [22], where once again, it shows promise in being more stable and efficient than the leading methods. The benefits AIS shows in these domains have transferred to the intrusion and anomaly detection domain in [23], where it is used to monitor computer networks with 99% accuracy.

Dendritic Cell Algorithm

DCA [2] is a subcategory of AIS that mimics the human immune system based on the danger theory. Dendritic Cells (DCs) are significant components of the immune system used to detect any foreign invader. Once there is a new intruder to the system, it is marked as either pathogen-associated molecular patterns (PAMP), flags as an anomalous behavior, Safe Signal, or Danger Signal. This can generate a warning as not safe and, over time, maybe determined as an anomaly. The DCA is comprised of the following four phases:

- Pre-processing Phase: Categorizes the data into three signal types (PAMP, Safe, Danger);
- Detection Phase: Concentration metrics are calculated for each Dendritic Cell (DC);
- Context Assessment Phase: Compares the concentration to a set threshold;
- Classification Phase: Classifies the DCs as anomalous or not.

The (DCA) [2] is combined with kernel density estimation (KDE) clustering for better detection of anomalies. This is the first research direction to combine AIS structure and clustering analysis in anomaly detection in the financial stock market data to the best of our knowledge. Measured by the Area Under the Curve (AUC) and False Alarm Ratio (FAR), this combined DCA-KDE approach has achieved a solution with AUC value that exceeds 0.84 and FAR values below 0.68. Although [2] is the first application of DCA in stock price manipulation, this approach has proven to be useful for similar anomaly detection problems in other domains. The authors in [15] applied DCA to the field of fault detection in Wind Turbines and showed an improvement in performance compared to different existing approaches with a FAR of less than 0.2 and an *F*-Score of over 0.93 for most datasets. In [16], a DCA approach is adopted for online error detection of robotic systems, where it showed the ability to detect errors in online data immediately. In [17], the DCA approach detects P2P bots for network security. They noted the method outperformed others due to decreased computational complexity and the ability to train on only regular network activity.

4. Clustering Analysis

Clustering is the grouping of data instances based on a set metric and/or threshold that ensures common instances are grouped to form a cluster. A clustering approach generates a set number of clusters, representing a certain class of the data. Different clustering-based anomalies detection approaches are used in many applications [3] [4] [13] [14]. Although there are many different clustering types in machine learning, four approaches have been implemented in this paper for anomaly detection. In this section, the four clustering algorithms discussed are; kernel density estimation (KDE), self-organizing maps (SOM), spectral clustering (SC), and density-based spatial clustering of applications with noise (DBSCAN).

4.1. Kernel Density Estimation (KDE)

KDE detects anomalies by comparing the density of a sample with its neighbors based on a kernel and set thresholds [24]. The algorithm in [24] uses a weighted neighborhood approach and proves its robustness in detecting anomalies with the neighborhood size parameter. An anomaly detection system that compares KDE to Entropy-based and PCA-based methods is presented in [25].

4.2. Self Organizing Maps (SOM)

SOMs are referred to as "Kohonen Neural Networks", a type of unsupervised learning based on competitive learning. They are typically used for classification or pattern recognition [26]. The basic algorithm is explained in [26], and, in summary, it is iteratively trained to find all necessary weight vectors that are eventually grouped based on their distance. After training is completed, the SOM is created and used for clustering. SOMs are efficiently used for anomaly detection [27].

4.3. Spectral Clustering

Spectral clustering is a graph-based clustering approach commonly used for anomaly detection with image-based data [28] to detect small-sized objects in hyperspectral images. In [29], the spectral clustering algorithm is used as an anomaly detection tool for wilderness search and rescue. It has been tested in the field by Texas A & M's Center for Emergency Informatics and has received positive feedback when used for search and rescue with UAV images.

4.4. Density Based Spatial Clustering of Applications with Noise

DBSCAN performs clustering by separating high and low-density regions within a data distribution. The DBSCAN algorithm is robust to noise and is highly scalable [30]; it is used to detect anomalies in traffic video surveillance. Similarly, DBSCAN is used along with KDE in [31] to identify abnormal moving speed for coastal surveillance systems. It is concluded that it can be integrated into coastal surveillance systems with a false alarm rate of zero.

5. The Proposed Hybrid Algorithm

The proposed hybrid model combines the AIS and clustering analysis capabilities. First, it performs data preprocessing, followed by incorporating the DCA algorithm. Once the first two stages of the DCA are completed, a clustering algorithm Ai is performed on the output of the DCA to detect anomalies in the dataset.

5.1. Phase 1: Dimension Reduction Using DCA

The first phase of the DCA Algorithm is used as a dimension reduction step that utilizes Principle Component Analysis (PCA) to reduce the original five dimensions into three dimensions. The three dimensions represent the three categories. Cp represents the PAMP category, Cs represents the Safe Signal category, and Cd represents the Danger category. Once the data are reduced and has a value for each category type, the algorithm moves to the detection phase to calculate the concentration of co-stimulation (C_{csm}), Semi-mature (C_{smDC}) and mature (C_{mDC}) for each DC using Equation (1) below.

$$C_{[csm,\ smDC,\ mDC]} = \frac{\left((Wp + Cp) + (Ws + Cs) + (Wd + Cd)\right)}{(Wp + Ws + Wd)} \tag{1}$$

where Wp, Ws, and Wd are the weights for the different categories of signal. The weights used in this paper are summarized in Table 1 [32]. Once the C_{csm}, C_{smDC}, and C_{mDC} are calculated, the new 3-dimensional dataset is passed to the clustering stage.

Table 1. Weights for DCA concentration function.

W	CSM	Semi-mature	Mature
PAMP signals (P)	2	1	2
Danger signals (D)	0	0	2
Safe signals (S)	2	1	−3

5.2. Phase 2: Anomaly Detection Using Clustering Analysis

In the second stage of the hybrid model, A clustering algorithm Ai uses the output of the DCA (*i.e.* the feature set of three dimensions, each representing a concentration of signal types) is introduced. We have used four different clustering algorithms, including KDE, DBSCAN, Spectral clustering, and SOM. Each of these algorithms works on datasets of different configurations and shapes. We have modified each algorithm to tailor our detection problem as follows:

5.2.1. KDE Detector

It focuses on the mean density of the data distribution to detect anomalies in the data. Such that clusters are created based on their difference from the mean density, which is recalculated each time a new cluster is created using Equation (2) and Equation (3). Where n is the number of data rows, g is a tuned smoothing parameter and Fi is the data instance. λ is used as a threshold to determine the cluster size for anomaly classification. The KDE detector algorithm is shown in **Figure 3**.

```
Algorithm: KDE-Anomaly Detector
Input: Dataset X of n records and d dimension, ClusterSizeLimit λ: the max size of anomaly clusters, gInc for the Mean Density expansion
Output: Top % outliers
Begin
F=X, initialize allClusters to empty
Loop
      Increase g for new Mean Density by gInc
     Calculate Mean Density for F with new g, (Eq. 2, Eq. 3)
          Loop for each record I for length of F
               If the difference between M and Fi is below g, add Fi to cluster and remove it from the set F
          Add the new cluster to the allClusters set
Until F is Empty
Return Top % Outliers by comparing clusters with λ
End
```

Figure 3. The KDE anomaly detector.

$$\text{Mean Density} = \frac{1}{ng}\sum_{i=1}^{n} K\left(\frac{f - Fi}{g}\right) \tag{2}$$

$$K(r) = \frac{1}{2\pi}\exp\left(-\frac{r^2}{2}\right) \tag{3}$$

5.2.2. SOM-Detector

The SOM for anomalies detection shown in **Figure 4** starts by training a model on the data, find the degree (number of mapped training vectors) of each SOM node, remove all nodes with degree less than the set threshold. Then, for each of remaining instances, it performs K-NN and calculates the mean distance to the nodes. Finally, it uses the mean distance or anomaly score to evaluate the data [33] [34] [35] [36].

```
Algorithm: SOM- Anomaly Detector
Input: Training Dataset X_Train, Evaluation Dataset X_Eval, Epsilion ε for Anomaly Score cut off value
Output: Top % outliers
Begin
     Initialize SOM Anomaly Detector, AD
     Train AD using X_Train dataset
     Evaluate X_Eval dataset to get Anomaly Score AS
     Compare AS scores against ε to classify as anomalous or not, classifier = AS > ε
End
Return Top % Outliers as True values in classifier array
```

Figure 4. The SOM anomaly detector.

5.2.3. Spectral-Detector

This method starts by finding a lower dimensional representation of the data that allows for more effective clustering using the DCA algorithm. The anomaly detector is created using the label diffusion approach of [37] and then it is combined with where Isolation Forests are used for the final clustering [38] as shown in **Figure 5**.

5.2.4. DBSCN Detector

The first step in **Figure 6** is to finds core samples of high density. It then uses a K-NN approach to calculate the required distances. Then an anomaly score for each sample is calculated. The samples are then flagged as anomalies when compared to a set threshold [39] [40].

5.2.5. CBLOF (*K*-Means or BIRCH)-Detector

In **Figure 7**, the (CBLOF) algorithm [33] is combined with the *K*-Means and BIRCH [34] algorithms. CBLOF assigns an anomaly score to the clustered data by first classifying each cluster as small or large using parameters alpha and beta. Secondly, the anomaly score is calculated based on cluster size and the distance to the nearest cluster classified as large.

Finally, **Figure 8** introduces the Hybrid AIS-Clustering anomaly detection, including the two phases of dimensionality reduction using DCA and then anomaly detection using clustering.

```
Algorithm: Spectral Clustering Anomaly Detector
Input: Dataset X of n records and d dimensions,
Epsilion ε for Anomaly Score cut off value
Output: Top % outliers
Begin
    Initialize Spectral Object, SO, and Transform the data with SO
    Initialize Spectral Anomaly Detector, AD
    Fit Dataset X to AD
    Apply decision function to X to get Anomaly Score AS
    Compare AS scores against ε to classify as anomalous or not, classifier = AS > ε
End
Return Top % Outliers as True values in classifier array
```

Figure 5. The spectral clustering anomaly detector.

```
Algorithm: DBSCAN Anomaly Detector
Input: Dataset X of n records and d dimensions,
Epsilion ε for Anomaly Score cut off value
Output: Top % outliers
Begin
    Import DBSCAN sklearn libraries
    Initialise DBSCAN Anomaly Detector, AD
    Fit Dataset X to AD
    Apply Optics function to X to get Anomaly Score AS
    Compare AS scores against ε to classify as anomalous or not, classifier = AS > ε
End
Return Top % Outliers as True values in classifier array
```

Figure 6. The DBSCAN anomaly detector.

```
Algorithm: CBLOF Anomaly Detector
Input: CC: Clusters from Clustering Algorithm, $\alpha$, $\beta$: coefficients for cluster size ratio
Output: Anomaly Scores AS
Begin
  For each cluster in the set CCs
    Classify CC as large or small based on $\alpha$ and $\beta$
    Calculate AS for each object in a big and large cluster
End
Return Anomaly Scores AS
```

Figure 7. The CBLOF anomaly detector.

```
Algorithm: Hybrid DCA-Ai
Input: Dataset X, clustering Algorithm Ai
Output: Top % outliers
Begin
Step1: X_DCA = DCA (X)
Step2: Choose the right detector based on the algorithm Ai
Step3: Feed X_DCA into Detector algorithm
End
Return Top % Outliers
```

Figure 8. The hybrid DCA-Ai algorithm.

6. Experimental Analysis

In this section, we have applied our proposed hybrid algorithm on five different stock market datasets. We have also compared the performance of the hybrid model against the individual-based models using various sets of evaluation metrics as shown next.

6.1. Data Collection

Financial data for the week of February 5th, 2018 through February 9th, 2018 was collected for each of the five stocks, including Amazon (AMZN), Apple (AAPL), Microsoft (MSFT), Intel Corp. (INTC), and Google (GOOGL) [2]. This date range was chosen as there were no reported manipulations during this time for the five stocks. The data was collected through Polygon.io API [40], and the extracted features are shown in Table 2.

Since each stock has different volumes in trading, the total size of each dataset differs from one another over a week of trading. The total size of each dataset is shown in Table 3.

Table 2. Features extracted from Polygon.io API.

Feature	Description
Timestamp	Epoch Timestamp
Sequence Number	Sequence Number
Exchange Id	Exchange Type
Size	Number of Shares
Price	Share Price
Conditions	Conditions on Trade

Table 3. Amount of data rows by stock.

Stock Ticker	Number of Data Rows
Amazon	590,387
AAPL	1,057,525
MSFT	818,603
INTC	571,101
GOOGL	225,349

6.2. Data Preprocessing

The final dataset used is comprised of five features based on the price of the stock and the timestamp of when the trades were made. The first feature, x_1 represents the share price after standard normalization techniques are applied. The second feature x_2, represents the share price after wavelet denoising is applied. Wavelet Denoising is a process that removes low-frequency components from the data, which is important when dealing with naturally high-frequency data such as stock market data. The wavelet denoising is applied to the stock price x, and the result is the second feature in the feature set [41] [42] [43], as shown in Figure 9. The third feature w is another calculated feature called the Wilson's Amplitude. Wilson's Amplitude is the sum of the number of times the difference between adjacent signals is above a set threshold p.

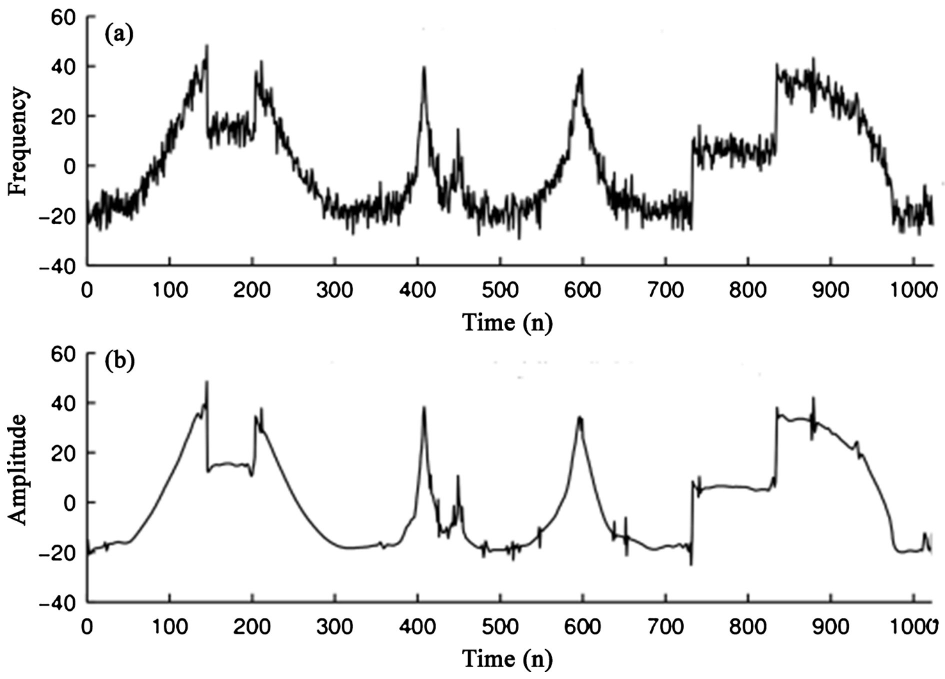

Figure 9. Example of wavelet denoising [43].

This is an effective feature to include as it is a good representation of an unusual increase or decrease of a stock price. Equation (4) and Equation (5) below are how to calculate Wilson's Amplitude.

$$s(t) = x(t) - x(t-1) \tag{4}$$

$$w(t) = \begin{cases} 3 * s(t), & s(t) > p \\ s(t), & s(t) \le p \end{cases} \tag{5}$$

where $x(t)$ is the price at time t and $x(t-1)$ is the previous price. Threshold p is calculated as the average value of $s(t)$. The final two features are Δx and Δw which are the rates of change in the stock price (x) and Wilson's Amplitude (w) over time respectively. In Equation (6), y can be replaced by x or w accordingly.

$$\Delta y(t) = \frac{y(t) - y(t-1)}{y(t-1)} * 100 \tag{6}$$

The final step of preprocessing is to apply standard normalization techniques to all feature values. The final set of features, $F = \{x, x_2, w, \Delta x, \Delta w\}$ are described in Table 4.

6.3. Evaluation Metrics

In this subsection we review some of the main evaluation measures that we have used to assess the quality of the proposed hybrid algorithms as well as those for the individual algorithm including, *F*1-Score, Sensitivity, Specificity, False Negative Rate (FNR), and False Alarm Rate (FAR). The final metric is the computation time which is the run time of each algorithm.

***F*-Score** is the weighted harmonic mean of the precision and recall defined below in Equation (9). Where *TP* is the number of True Positives, *FP* is the number of False Positives, *FN* is the number of False Negatives and *TN* is the number of True Negatives.

$$\text{Precision} = \left(\frac{TP}{TP + FP}\right) \tag{7}$$

$$\text{Recall} = \left(\frac{TP}{TP + FN}\right) \tag{8}$$

$$\text{F-Score} = 2 * \left(\frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}\right) \tag{9}$$

Table 4. Final set of input features.

Feature	Description
x	Share Price (Normalized)
x_2	Wavelet Denoised Share Price
w	Wilson's Amplitude
Δx	Change in Stock Price Over Time
Δw	Change in Wilson's Amplitude Over Time

Accuracy is a measure of how often data are classified correctly.

$$\text{Accuracy} = \left(\frac{TP + TN}{TP + TN + FP + FN}\right) \tag{10}$$

Sensitivity is a measure of the proportion of True Positive cases that got properly classified as positive.

$$\text{Sensitivity} = \left(\frac{TP}{TP + FN}\right) \tag{11}$$

Specificity is a measure of the proportion of True Negative cases that got properly classified as negative.

$$\text{Specificity} = \left(\frac{TN}{TN + FP}\right) \tag{12}$$

False Negative Rate (FNR) is the proportion of falsely classified negatives over all expected positive cases.

$$\text{FNR} = \left(\frac{FN}{FN + TP}\right) \tag{13}$$

False Alarm Rate (FAR) is the proportion of falsely classified positives over all expected negative cases.

$$\text{FPR} = \left(\frac{FP}{FP + TN}\right) \tag{14}$$

Area Under the Curve (AUC) is a measure of separability for classification and is used to determine how well a model can distinguish the different classes. The closer the value is to 1 then the better the model is at distinguishing separate classes. The value is determined by taking the area under the curve created when plotting FPR (*x*-axis) vs. TPR (*y*-axis).

6.4. Experimental Results

The following section compares the hybrid DCA-KDE, DCA-SOM, DCA-DBSCAN, DCA-Spectral, DCA-CBLOF (*K*-Means) and DCA-CBLOF (BIRCH) algorithms along with their individual detection approach. In Figure 10, we can see that hybrid DCA-KDE has the best performance across all datasets in terms of *F*-Score. The performance of the CBLOF (BIRCH) and CBLOF (*K*-Means) increases when combined with DCA. DCA-CBLOF (*K*-Means) achieves the best improvement on Amazon, Apple, and Google datasets. The DCA-CBLOF (BIRTH) has its best performance for the Amazon and Apple datasets. SOM does significantly change the *F*-Score when combined with DCA, except some inconsistency on the MSFT dataset. DCA-DBSCAN performed the worst, and DCA-Spectral did not achieve much benefit and continue to perform poorly in terms of *F*-Score.

Figure 11 compares the accuracy of each algorithm before and after DCA is applied. Once again, we can observe that KDE benefits the most when combined with DCA. As the SOM has a strong performance as an individual detector, the combination of DCA-SOM does achieve a substantial increase in the accuracy. Both CBLOF (BIRCH) and CBLOF (*K*-Means) perform similarly in terms of accuracy, achieving minor improvements when combined with DCA. DCA-DBSCAN has

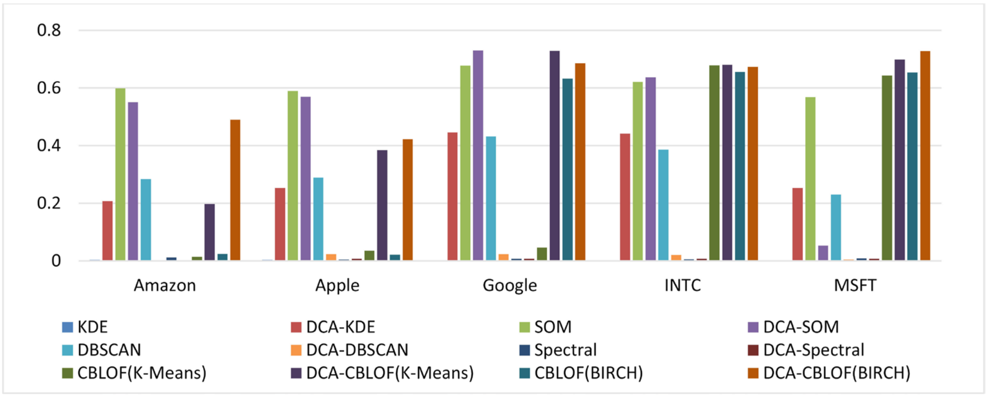

Figure 10. Individual vs. hybrid anomaly detectors (*F*-Score).

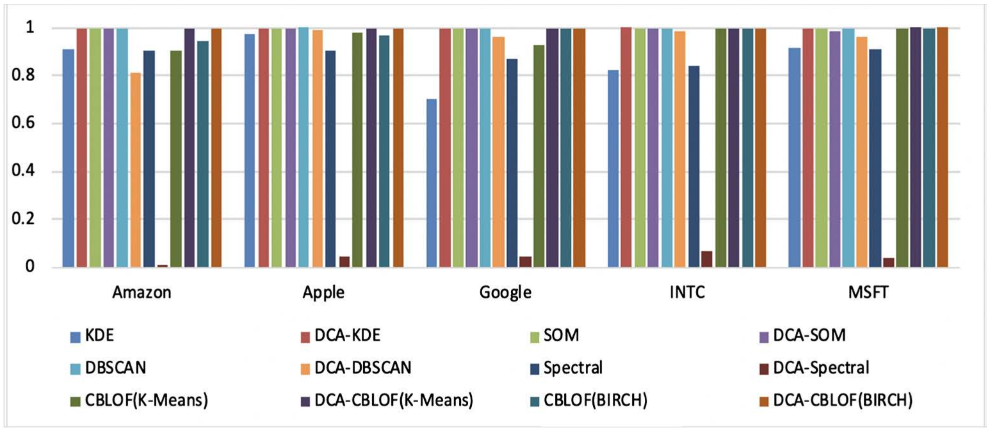

Figure 11. Individual vs. hybrid anomaly detectors (accuracy).

obtained a drop-in accuracy across all datasets, with some being minuscule, as seen on Apple and INTC datasets. DCA-Spectral has the worst performance with a large drop in accuracy. Note that the accuracy can be close to one hundred percent when an algorithm performs well due to the limited number of anomalies within a large dataset. Similarly, in Figure 14, due to the small number of anomalies in the large dataset, for well performing algorithms the sensitivity can be close to 100%.

In Figure 12, DCA-KDE has the highest sensitivity factor across all datasets, while DCA-SOM observes small improvement. Similarly, to DCA-SOM, the CBLOF (*K*-Means) and CBLOF (BIRCH) achieves minor enhancements, with the exception of CBLOF (BIRCH) on the Google dataset. The DCA-DBSCAN shows some inconsistency across datasets with minor improvements on Apple and Amazon and negative effects on MSFT. The DCA-Spectral achieved a large increase in sensitivity across all datasets. Figure 13 compares the specificity of each algorithm. We can see a similar trend as in Figure 12, such that the DCA-KDE has the best performance, the DCA-SOM is consistent, and the DCA-DBSCAN observes a small drop in specificity across all datasets. The CBLOF (*K*-Means) and the CBLOF (BIRCH) have the same performance, with CBLOF (*K*-Means) has a slightly larger benefactor. Although the DCA-Spectral has obtained a large increase in sensitivity as shown in Figure 13, it has negative results in terms of specificity.

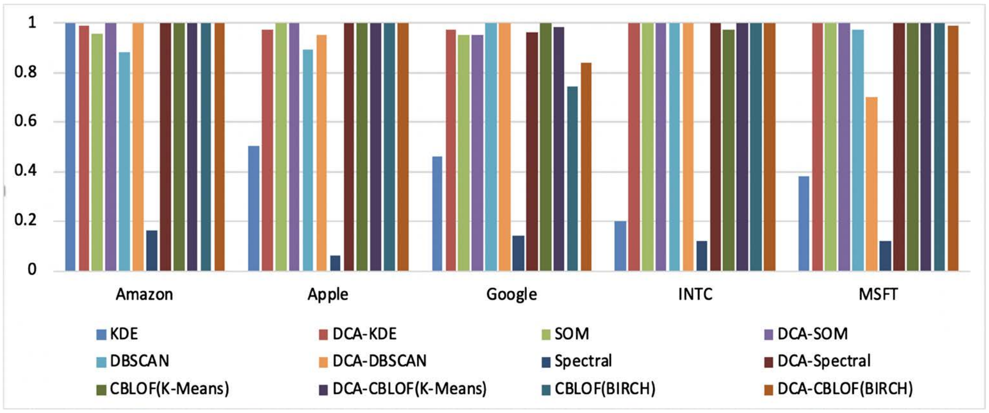

Figure 12. Individual vs. hybrid anomaly detectors (sensitivity).

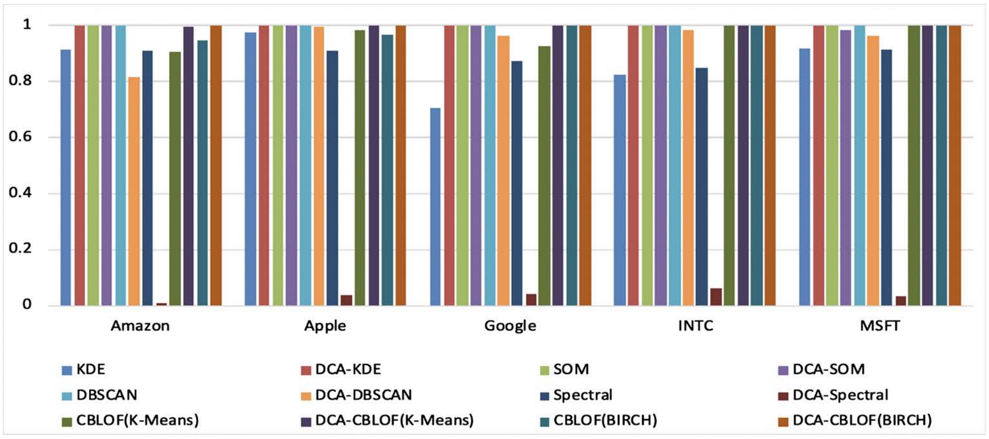

Figure 13. Individual vs. hybrid anomaly detectors (specificity).

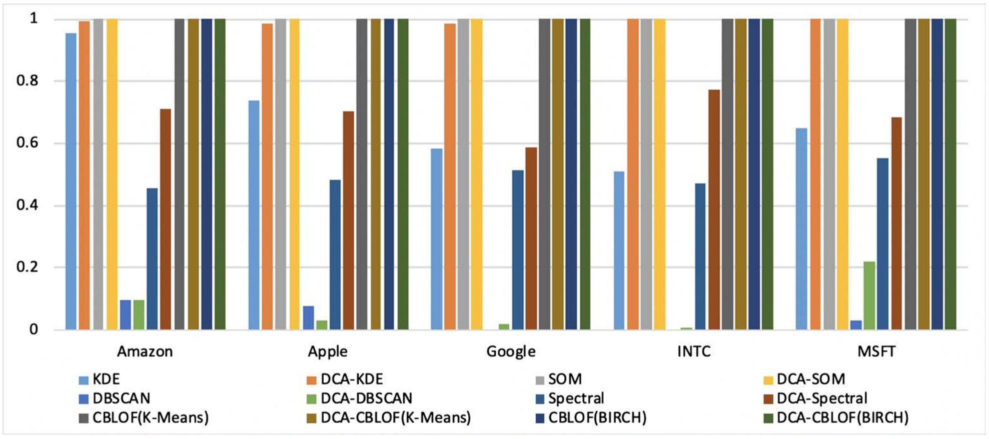

Figure 14. Individual vs. hybrid anomaly detectors (AUC).

Figure 14 shows that the individual-based SOM has the best performing algorithm in terms of the area under the curve metric, and the DBSCAN is the worst performing individual-based clustering algorithm. Similarly, once combined with

DCA in the hybrid model, DCA-SOM, DCA-BIRCH, and DCA-CBLOF (*K*-Means) are the best performing algorithms, and DCA-DBSCAN performed the worst. Noting some inconsistency in the DBSCAN/DCA-DBSCAN results, overall, the combination of DCA with the individual-based algorithms has shown positive improvement. It is also worth mentioning that the SOM performed well as an individual-based detector, and therefore, it only obtained a marginal improvement.

In Table 5, in general, combining the DCA with the individual-based algorithms decreases the false-negative rate. The DCA-KDE and DCA-Spectral have achieved the largest improvement. The DCA-SOM, DCA-CBLOF (*K*-Means), DCA-CBLOF (BIRCH) have obtained an improvement in the false-negative rate. However, due to the strong performance on the individual-based methods, the benefit was not as significant.

Table 6 shows the effect of the hybrid model in terms of the false alarm rate. Compared to previous metrics, the FAR has the least consistent effect across the hybrid clustering algorithms. KDE and its hybrid combination show a great improvement across all datasets. The DCA-CBLOF (*K*-Means) and DCA-CBLOF (BIRCH) have the best FAR values on Apple and Amazon datasets. DCA-SOM once again has minuscule improvement, but it is noted that it does not negatively affect this result. DCA-Spectral had a negative response with the false alarm increasing substantially, and DCA-DBSCAN showed inconsistency across the datasets. Table 7 shows the computation time of the hybrid models and individual-based algorithms. Overall, the hybrid model decreases the computation time across all algorithms. The biggest benefactors are DCA-KDE and DCA-DBSCAN. DCA-Spectral showed minimal improvement for Amazon, MSFT, and INTC datasets. The SOM, once again, did not as much improvement as other approaches. DCA-CBLOF (BIRCH) and DCA-CBLOF (*K*-Means) showed a slight increase in the time.

Table 5. Individual vs. hybrid anomaly detectors (FNR).

	Amazon	Apple	Google	INTC	MSFT
KDE	0	0.5	0.54	0.8	0.62
DCA-KDE	0.01	0.03	0.03	0	0
SOM	0.042	0.004	0.048	0.001	0.003
DCA-SOM	0	0	0.05	0	0
DBSCAN	0.12	0.109	0	0	0.03
DCA-DBSCAN	0	0.05	0	0	0.3
Spectral	0.84	0.94	0.86	0.88	0.88
DCA-Spectral	0	0	0.04	0	0
CBLOF (*K*-Means)	0	0	0	0.03	0
DCA-CBLOF (*K*-Means)	0	0	0.02	0	0
CBLOF (BIRCH)	0	0	0.26	0	0
DCA-CBLOF (BIRCH)	0	0	0.16	0	0.01

Table 6. Individual vs. hybrid anomaly detectors (FAR).

	Amazon	Apple	Google	INTC	MSFT
KDE	0.0891	0.0283	0.2955	0.1796	0.0828
DCA-KDE	0.0013	0.0005	0.0011	0.0004	0.0007
SOM	0.0008	0.0005	0.0015	0.0009	0.0007
DCA-SOM	0.0011	0.0005	0.0012	0.0008	0.0174
DBSCAN	0.0007	0.0004	0.0012	0.0006	0.0008
DCA-DBSCAN	0.1879	0.0072	0.0386	0.0167	0.0384
Spectral	0.0917	0.0904	0.1296	0.1544	0.0882
DCA-Spectral	0.9927	0.9621	0.9589	0.9367	0.9664
CBLOF (*K*-Means)	0.0959	0.0191	0.0746	0.0006	0.0005
DCA-CBLOF (*K*-Means)	0.0055	0.0011	0.0012	0.0007	0.0004
CBLOF (BIRCH)	0.0554	0.0329	0.0011	0.0007	0.0005
DCA-CBLOF (BIRCH)	0.0014	0.0009	0.0011	0.0007	0.0004

Table 7. Individual vs. hybrid anomaly detectors (computation time).

	Amazon	Apple	Google	INTC	MSFT
KDE	268	450	120	321	750
DCA-KDE	17	100	15	25	24
SOM	13	22	6	14	18
DCA-SOM	12	20	5	12	17
DBSCAN	21,108	62,246	3042	20,737	36,814
DCA-DBSCAN	3131	10,125	453	2850	5678
Spectral	8748	8408	18,770	8493	8499
DCA-Spectral	8432	8326	8322	4980	8328
CBLOF (*K*-Means)	28	41	7	20	30
DCA-CBLOF (*K*-Means)	26	46	7	22	35
CBLOF (BIRCH)	20	42	10	16	36
DCA-CBLOF (BIRCH)	22	46	10	15	40

Figure 15 shows the anomalies in the data in red compared to all other data points in black. The diagram on the left is the actual anomalies in the data where the diagram on the right is the data points flagged as anomalies. We can see from Figure 16 that the top right anomalies are the easier type of stock manipulation to detect, and the others are more hidden within the actual data. Figure 16 illustrates the application of the DCA to decreases the feature space of the data. SOM and DCA-SOM are used for Figure 15 and Figure 16, respectively, as examples.

Tables 8-11 show the % of improvements of using hybrid models against the individual-based algorithm measured by the *F*-Score, accuracy, sensitivity, and specificity, respectively. We can observe that the DCA-KDE, DCA-CBLOF (*K*-Means),

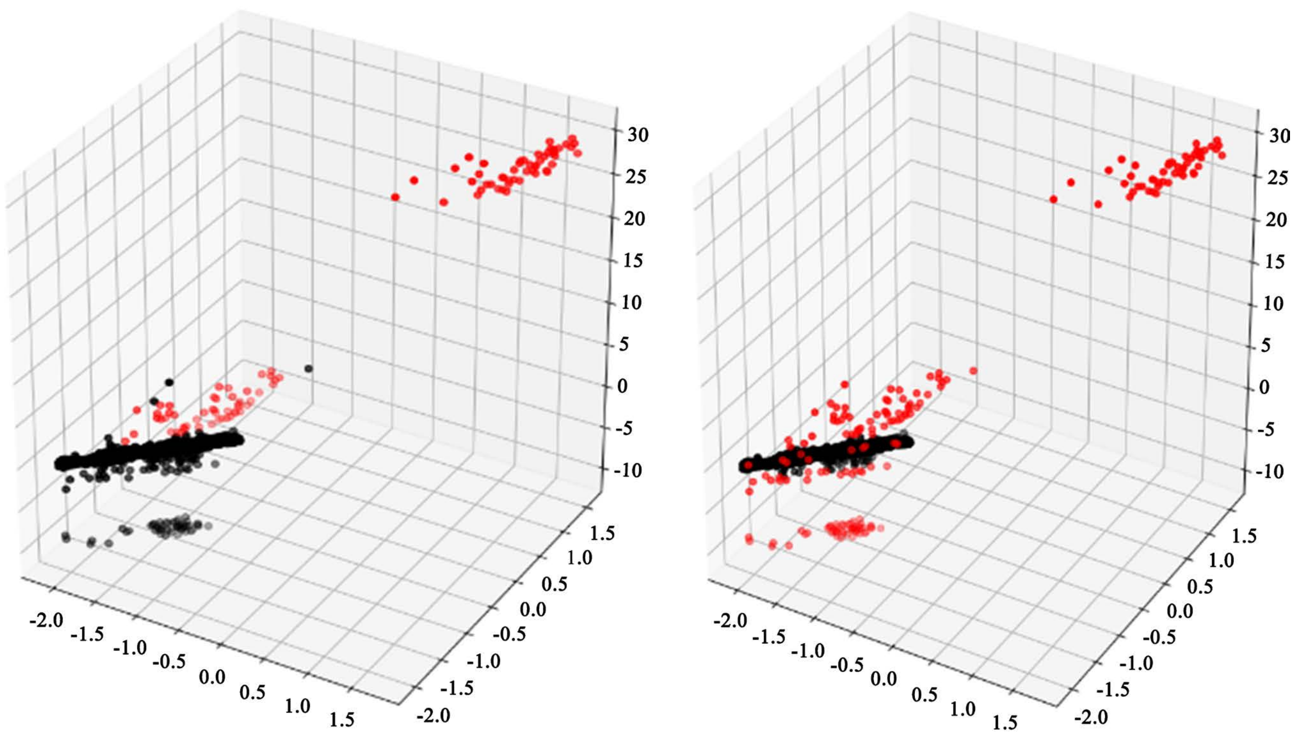

Figure 15. True anomalies vs. detected anomalies for (SOM) (*x*, *y*, *z*).

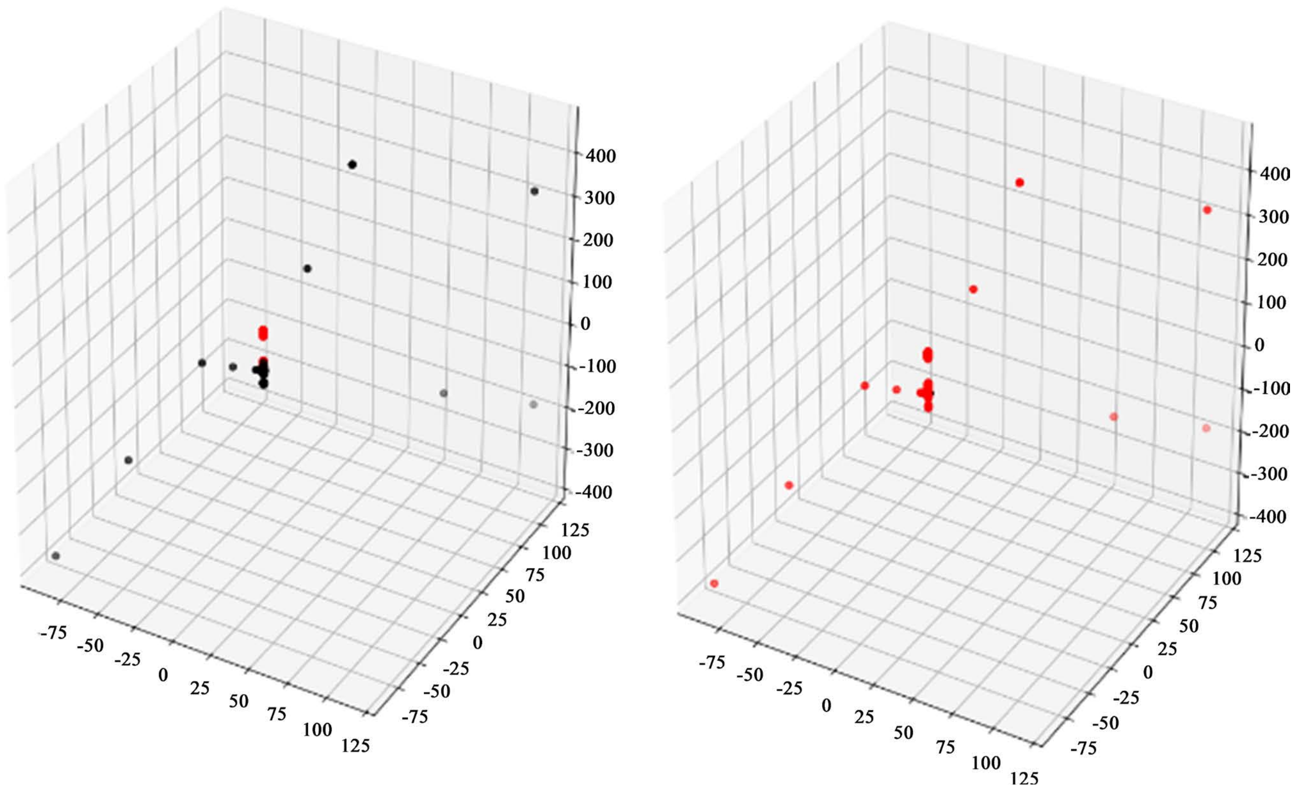

Figure 16. True anomalies vs. detected anomalies for DCA-SOM model (*x*, *y*, *z*).

DCA-CBLOF (BIRCH) has the greatest *F*-Score improvement as shown in Table 8. It is also observed that DCA-DBSCAN and DCA-Spectral do not always see the same positive result. Due to the strong performance of SOM individually, the percentage change is small for most datasets. In Table 9, both DCA-CBLOF (*K*-Means), and DCA-CBLOF (BIRCH) obtain spikes in accuracy across all datasets but a minor decrease on the Intel datasets for the DCA-CBLOF (*K*-Means). DCA-Spectral has poor results of less than 90 percent reduction. Table 10 shows the inconsistency of DCA-DBSCAN, a minor improvement on the SOM, and poor performance with spectral clustering. DCA-CBLOF (*K*-Means), DCA-CBLOF (BIRCH) obtain similar improvement as DCA-SOM with DCA-SOM is consistent across datasets. Similar observations can be made in Table 11.

Table 12 shows the overall improvement the hybrid model has in terms of the AUC metric. The only notably negatively affected were Amazon and Apple for DBSCAN, which demonstrates the DBSCAN approach's inconsistency. Table 13

Table 8. Percentage of improvement (*F*-Score).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	98.18	98.78	99.65	99.91	99.57
DCA-SOM vs SOM	8.775	3.548	7.212	2.461	99.99
DCA-DBSCAN vs DBSCAN	−99.99	−99.99	−99.99	−99.99	−99.99
DCA-Spectral vs Spectral	−99.99	38.15	−5.557	28.44	−26.96
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	92.94	90.80	93.76	0.287	8.039
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	95.13	95.05	7.764	2.623	10.21

Table 9. Percentage of improvement (accuracy).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	8.797	2.788	29.49	17.94	8.219
DCA-SOM vs SOM	0.024	0.0006	0.0363	0.0055	1.696
DCA-DBSCAN vs DBSCAN	−23.04	−0.6823	−3.894	−1.642	−3.915
DCA-Spectral vs Spectral	−99.99	−99.99	−99.99	−99.99	−99.99
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	9.091	1.806	7.326	−0.001	0.012
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	5.406	3.194	0.016	0.006	0.016

Table 10. Percentage of improvement (sensitivity).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	−1	48.31	52.58	80.00	62.00
DCA-SOM vs SOM	4.200	0.400	−0.2105	0.100	0.301
DCA-DBSCAN vs DBSCAN	12.00	5.747	0	0	−38.57
DCA-Spectral vs Spectral	84.00	94.00	85.42	88.00	88.00
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	0	0	−2.041	3.000	0
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	0	0	11.90	0	−1

Table 11. Percentage of improvement (specificity).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	8.799	2.784	29.48	17.73	8.213
DCA-SOM vs SOM	−0.0265	−0.0008	0.0367	0.0054	−1.697
DCA-DBSCAN vs DBSCAN	−23.05	−0.683	−3.896	−1.642	−3.912
DCA-Spectral vs Spectral	−99.99	−99.99	−99.99	−99.99	−99.99
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	9.098	1.807	7.343	−0.004	0.0122
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	5.410	3.195	−0.002	0.006	0.016

Table 12. Percentage of improvement (AUC).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	3.916	25.18	40.86	48.97	35.12
DCA-SOM vs SOM	0.0017	0.0003	0.002	0.0015	0.0020
DCA-DBSCAN vs DBSCAN	−1.463	−99.99	96.98	96.67	86.05
DCA-Spectral vs Spectral	35.82	31.71	12.81	39.24	19.14
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	0.0003	0.0002	0.002	0.011	0.0014
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	0.0001	0.0001	0.0018	−0.0001	0.0011

Table 13. Percentage of improvement (FNR).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	1	−99.99	−99.99	−1	−1
DCA-SOM vs SOM	−100	−100	4.000	−100	−100
DCA-DBSCAN vs DBSCAN	−100	−100	0	0	90
DCA-Spectral vs Spectral	−100	−100	−99.99	−100	−100
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	0	0	100	−100	0
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	0	0	−62.50	0	100

shows the positive results of the hybrid model in terms of FNR. Almost all approaches across all datasets showed a strong result, with very few false negatives being detected after combining with AIS. Table 14 shows the outcome in terms of FAR, which shows the most substantial benefactor as the DCA-KDE. We can also see that DCA-DBSCAN and DCA-Spectral had a significant increase in the false alarm rate. Both CBLOF models achieve the greatest benefit of combining with DCA in terms of FAR. The results are not as consistently reliable across all datasets like DCA-KDE but have an overall positive affect. Table 15 shows the improvement in terms of computation time. The hybrid combination positively reduced the computational time for all algorithms with DCA-KDE and DCA-DBSCAN being the largest benefactors. There is some increase in computation time for the CBLOF models, but the difference is only a minuscule amount (<5 s).

Table 14. Percentage of improvement (FAR).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	−99.99	−99.99	−99.99	−99.99	−99.99
DCA-SOM vs SOM	23.99	1.495	−31.69	−6.754	95.76
DCA-DBSCAN vs DBSCAN	99.61	94.76	96.98	96.67	97.95
DCA-Spectral vs Spectral	90.76	90.60	86.49	83.51	90.87
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	−99.99	−99.99	−99.99	5.319	−29.07
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	−99.99	−99.99	1.639	−8.247	−45.21

Table 15. Percentage of improvement (computation time).

	Amazon	APPL	GOOGL	INTC	MSFT
DCA-KDE vs. KDE	−93.66	−77.78	−87.50	−92.21	−96.80
DCA-SOM vs SOM	−7.69	−9.09	−16.67	−14.29	−5.560
DCA-DBSCAN vs DBSCAN	−85.17	−83.73	−85.11	−86.26	−84.58
DCA-Spectral vs Spectral	−3.610	−0.980	−55.66	−41.36	−2.01
CBLOF (*K*-Means) vs DCA-CBLOF (*K*-Means)	−7.143	12.19	0	10	16.67
CBLOF (BIRCH) vs DCA-CBLOF (BIRCH)	10	9.524	0	−6.25	11.11

7. Conclusion and Future Directions

The effect of combining clustering algorithms with DCA to create a hybrid model differed between clustering approaches is discussed in this project. The DCA-KDE has the biggest benefactor of the hybrid combination. KDE, as the individual-based solution did not perform well, but once combined with DCA, it demonstrated positive results. The comparable solutions were DCA-SOM, DCA-CBLOF (*K*-Means), and DCA-CBLOF (BIRCH). It has been observed in this project that the SOMs model is effective methods to model financial time series data and can be improved even further when combined with DCA. This has been a very encouraging section of this research as it is believed that SOMs have not been applied to anomaly detection in financial data and may be used as a reliable tool to do so. The DCA-CBLOF (*K*-Means) showed a significant improvement in *F*-Score and FAR and was also one of the best algorithms in terms of AUC. DCA-CBLOF (BIRCH) performed similarly to DCA-CBLOF (*K*-Means), as it achieves the most considerable improvement in *F*-Score and FAR. However, DCA-CBLOF (BIRCH) did show some minor improvement across certain datasets. The DCA-DBSCAN shows that not all clustering algorithms will have clear improvement across certain metrics. DBSCAN also had the most inconsistency across datasets, which demonstrates that it may not be able to handle financial time series data and the other approaches. DBSCAN illustrates the advantage of the hybrid model in computation time as it saw a significant decrease. Spectral saw the greatest benefit of combining with DCA in a hybrid model in terms of AUC and FNR. The FNR dropped significantly across all datasets but at the cost of a higher FAR. In this paper, we have introduced the adoption of the KDE, SOM, *K*-Means, and BIRCH as approaches to not only to clustering data but also to achieve comparable results and even exceeded KDE in AUC and FAR when combined with the DCA. These hybrid combinations had advantages in terms of the compared metrics without a large penalty in computation time and proved to be a competitive approach to DCA-KDE. In summary, this project has investigated the hybrid model of DCA-Ai with multiple standard clustering approaches. It has found that DCA-SOM, DCA-CBLOF (*K*-Means), and DCA-CBLOF (BIRCH) can be an effective tool for anomaly detection in the financial stock market data and is a competitive solution to the leading KDE approach that inspired this paper. It is believed the application of SOMs, CBLOF (*K*-Means), and CBLOF (BIRCH) for anomaly detection has not been heavily researched in this domain until now and shows promise and opportunity to do so. Possible future directions include expanding the datasets and possibly analyzing the commonalities between the clustering algorithms that have a positive effect when combined with DCA versus algorithms that are negatively affected. We will also investigate the combination of new clustering methods with DCA to find possible new competitive hybrid models for anomaly detection. Also, a more in-depth analysis of which of the two types of manipulation attacks was easier to find and how each algorithm performed separately could yield interesting results. These future works would allow us to understand which combination is the strongest for anomaly detection since it can be difficult to distinguish, which is the best between the top-performing models.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Leangarun, T., Tangamchit, P. and Thajchayapong, S. (2018) Stock Price Manipulation Detection using Generative Adversarial Networks. 2018 *IEEE Symposium Series on Computational Intelligence* (*SSCI*), Bangalore, 18-21 November 2018, 2104-2111. https://doi.org/10.1109/SSCI.2018.8628777

[2] Rizvi, B., Belatreche, A. and Bouridane, A. (2019) A Dendritic Cell Immune System Inspired Approach for Stock Market Manipulation Detection. 2019 *IEEE Congress on Evolutionary Computation* (*CEC*), Wellington, 10-13 June 2019, 3325-3332. https://doi.org/10.1109/CEC.2019.8789938

[3] Ahmed, M., Choudhury, N. and Uddin, S. (2017) Anomaly Detection on Big Data in Financial Markets. 2017 *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (*ASONAM*), Sydney, July 2017, 998-1001. https://doi.org/10.1145/3110025.3119402

[4] Cao, Y., Li, Y., Coleman, S., Belatreche, A. and McGinnity, T.M. (2014) Detecting Price Manipulation in the Financial Market. 2014 *IEEE Conference on Computational Intelligence for Financial Engineering & Economics* (*CIFEr*), London, 27-28 March 2014, 77-84. https://doi.org/10.1109/CIFEr.2014.6924057

[5] Golmohammadi, K. and Zaiane, O.R. (2015) Time Series Contextual Anomaly Detection for Detecting Market Manipulation in Stock Market. 2015 *IEEE International Conference on Data Science and Advanced Analytics* (*DSAA*), Paris, 19-21 October 2015, 1-10. https://doi.org/10.1109/DSAA.2015.7344856

[6] Fan, S.S., Liu, G.H. and Chen, Z. (2017) Anomaly Detection Methods for Bankruptcy Prediction. 2017 4*th International Conference on Systems and Informatics* (*ICSAI*), Hangzhou, 11-13 November 2017, 1456-1460. https://doi.org/10.1109/ICSAI.2017.8248515

[7] Raza, M. and Qayyum, U. (2019) Classical and Deep Learning Classifiers for Anomaly Detection. 2019 16*th International Bhurban Conference on Applied Sciences and Technology* (*IBCAST*), Islamabad, 8-12 January 2019, 614-618. https://doi.org/10.1109/IBCAST.2019.8667245

[8] Samarakoon, P.A. and Athukorala, D.A.S. (2017) System Abnormality Detection in Stock Market Complex Trading Systems Using Machine Learning Techniques. 2017 *National Information Technology Conference* (*NITC*), Colombo, 14-15 September 2017, 125-130. https://doi.org/10.1109/NITC.2017.8285660

[9] Anandharaj, A. and Sivakumar, P.B. (2019) Anomaly Detection in Time Series Data Using Hierarchical Temporal Memory Model. 2019 3*rd International Conference on Electronics*, *Communication and Aerospace Technology* (*ICECA*), Coimbatore, 12-14 June 2019, 1287-1292. https://doi.org/10.1109/ICECA.2019.8821966

[10] Cao, Y., Li, Y., Coleman, S., Belatreche, A. and McGinnity, T.M. (2015) Adaptive Hidden Markov Model with Anomaly States for Price Manipulation Detection. *IEEE Transactions on Neural Networks and Learning Systems*, **26**, 318-330. https://doi.org/10.1109/TNNLS.2014.2315042

[11] Liu, J.M., Tian, J., Cai, Z.X., Zhou, Y., Luo, R.H. and Wang, R.R. (2017) A Hybrid Semi-Supervised Approach for Financial Fraud Detection. 2017 *International Conference on Machine Learning and Cybernetics* (*ICMLC*), Ningbo, 9-12 July 2017, 217-222. https://doi.org/10.1109/ICMLC.2017.8107767

[12] Li, M., Kashef, R. and Ibrahim, A. (2020) Multi-Level Clustering-Based Outlier's Detection (MCOD) Using Self-Organizing Maps. *Big Data and Cognitive Computing*, **4**, 24. https://doi.org/10.3390/bdcc4040024

[13] Al-Thani, H., Hassen, H., Al-Maadced, S., Fetais, N. and Jaoua, A. (2018) Unsupervised Technique for Anomaly Detection in Qatar Stock Market. 2018 *International Conference on Computer and Applications* (*ICCA*), Beirut, 25-26 August 2018, 116-119. https://doi.org/10.1109/COMAPP.2018.8460282

[14] Camino, R.D., State, R., Montero, L. and Valtchev, P. (2017) Finding Suspicious Activities in Financial Transactions and Distributed Ledgers. 2017 I*EEE International Conference on Data Mining Workshops* (*ICDMW*), New Orleans, 18-21 November 2017, 787-796. https://doi.org/10.1109/ICDMW.2017.109

[15] Alizadeh, E., Meskin, N. and Khorasani, K. (2018) A Dendritic Cell Immune System Inspired Scheme for Sensor Fault Detection and Isolation of Wind Turbines. *IEEE Transactions on Industrial Informatics*, **14**, 545-555. https://doi.org/10.1109/TII.2017.2746761

[16] Mokhtar, M., Bi, R., Timmis, J. and Tyrrell, A.M. (2009) A modified Dendritic Cell Algorithm for On-Line Error Detection in Robotic Systems. 2009 IEEE Congress on Evolutionary Computation, Trondheim, 18-21 May 2009, 2055-2062. https://doi.org/10.1109/CEC.2009.4983194

[17] Wang, L. and Fang, X.J. (2015) The Detection of P2P Bots Using the Dendritic Cells Algorithm. 2015 Int*ernational Conference on Estimation*, *Detection and Informa-*

tion Fusion (*ICEDIF*), Harbin, 10-11 January 2015, 299-302. https://doi.org/10.1109/ICEDIF.2015.7280211

[18] Huang, Y., Chung, W. and Tang, X. (2018) A Temporal Recurrent Neural Network Approach to Detecting Market Anomaly Attacks. 2018 *IEEE International Conference on Intelligence and Security Informatics* (*ISI*), Miami, 9-11 November 2018, 160-162. https://doi.org/10.1109/ISI.2018.8587397

[19] De Mello Honorio, L., Da Silva, A.M.L. and Barbosa, D.A. (2012) A Cluster and Gradient-Based Artificial Immune System Applied in Optimization Scenarios. *IEEE Transactions on Evolutionary Computation*, **16**, 301-318. https://doi.org/10.1109/TEVC.2010.2044242

[20] Freitas, A.A. and Timmis, J. (2007) Revisiting the Foundations of Artificial Immune Systems for Data Mining. *IEEE Transactions on Evolutionary Computation*, **11**, 521-540. https://doi.org/10.1109/TEVC.2006.884042

[21] Kashef, R.F. (2018) Proceedings of the KDD 2017: Workshop on Anomaly Detection in Finance, PMLR 71. 43-55.

[22] Deng, Y., Bao, F., Kong, Y., Ren, Z. and Dai, Q. (2017) Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. *IEEE Transactions on Neural Networks and Learning Systems*, **28**, 653-664. https://doi.org/10.1109/TNNLS.2016.2522401

[23] Leangarun, T., Tangamchit, P. and Thajchayapong, S. (2016) Stock Price Manipulation Detection Using a Computational Neural Network Model. 2016 *Eighth International Conference on Advanced Computational Intelligence* (*ICACI*), Chiang Mai, 14-16 February 2016, 337-341. https://doi.org/10.1109/ICACI.2016.7449848

[24] Hu, W.M., Gao, J., Li, B., Wu, O., Du, J.P. and Maybank, S. (2020) Anomaly Detection Using Local Kernel Density Estimation and Context-Based Regression. *IEEE Transactions on Knowledge and Data Engineering*, **32**, 218-233. https://doi.org/10.1109/TKDE.2018.2882404

[25] Ahmed, T. (2009) Online Anomaly Detection Using KDE. *GLOBECOM* 2009—2009 *IEEE Global Telecommunications Conference*, Honolulu, 30 November-4 December 2009, 1-8, https://doi.org/10.1109/GLOCOM.2009.5425504

[26] Aquize, V.G., Emery, E. and De Lima Neto, F.B. (2017) Self-Organizing Maps for Anomaly Detection in Fuel Consumption. Case Study: Illegal Fuel Storage in Bolivia. 2017 *IEEE Latin American Conference on Computational Intelligence* (*LA-CCI*), *Arequipa*, 8-10 November 2017, 1-6. https://doi.org/10.1109/LA-CCI.2017.8285697

[27] Sukkhawatchani, P. and Usaha, W. (2008) Performance Evaluation of Anomaly Detection in Cellular Core Networks Using Self-Organizing Map. 2008 5*th International Conference on Electrical Engineering*/*Electronics*, *Computer*, *Telecommunications and Information Technology*, Krabi, 14-17 May 2008, 361-364. https://doi.org/10.1109/ECTICON.2008.4600446

[28] Ben Salem, M., Ettabaâ, K.S. and Bouhlel, M.S. (2016) Anomaly Detection in Hyperspectral Images Based Spatial Spectral Classification. 2016 7*th International Conference on Sciences of Electronics*, *Technologies of Information and Telecommunications* (*SETIT*), Hammamet, 18-20 December 2016, 166-170. https://doi.org/10.1109/SETIT.2016.7939860

[29] Proft, J., Suarez, J. and Murphy, R. (2015) Spectral Anomaly Detection with Machine Learning for Wilderness Search and Rescue. 2015 *IEEE MIT Undergraduate Research Technology Conference* (*URTC*), Cambridge, 7-8 November 2015, 1-3. https://doi.org/10.1109/URTC.2015.7563746

[30] Ranjith, R., Athanesious, J.J. and Vaidehi, V. (2015) Anomaly Detection Using

DBSCAN Clustering Technique for Traffic Video Surveillance. 2015 7*th International Conference on Advanced Computing* (*ICoAC*), Chennai, 15-17 December 2015, 1-6. https://doi.org/10.1109/ICoAC.2015.7562795

[31] Loi, N.V., Trung Kien, T., Hop, T.V., Thanh Son, L. andKhuong, N.V. (2020) Abnormal Moving Speed Detection Using Combination of Kernel Density Estimator and DBSCAN for Coastal Surveillance Radars. 2020 7*th International Conference on Signal Processing and Integrated Networks* (*SPIN*), Noida, 27-28 February 2020, 143-147. https://doi.org/10.1109/SPIN48934.2020.9070885

[32] Elisa, N., Yang, L.Z. and Naik, N. (2018) Dendritic Cell Algorithm with Optimised Parameters Using Genetic Algorithm. 2018 *IEEE Congress on Evolutionary Computation* (*CEC*), Rio de Janeiro, 8-13 July 2018, 1-8. https://doi.org/10.1109/CEC.2018.8477932

[33] Ullah, I., Hussain, H., Ali, I. and Liaquat, A. (2019) Churn Prediction in Banking System Using K-Means, LOF, and CBLOF. 2019 *International Conference on Electrical, Communication, and Computer Engineering* (*ICECCE*), Swat, 24-25 July 2019, 1-6. https://doi.org/10.1109/ICECCE47252.2019.8940667

[34] Zhao, Y., Nasrullah, Z. and Li, Z. (2019) PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research*, **20**, 1-7.

[35] Tian, J., Azarian, M.H. and Pecht, M. (2014) Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm. European Conference of the Prognostics and Health Management Society.

[36] Das, S., Moore, T., Wong, W.-K., Stumpf, S., Oberst, I., McIntosh, K. and Burnett, M. (2013) End-User Feature Labeling: Supervised and Semi-Supervised Approaches Based on Locally-Weighted Logistic Regression. *Artificial Intelligence*, **204**, 56-74. https://doi.org/10.1016/j.artint.2013.08.003

[37] Das, S. (2018) Ad_examples/ad/spectral_outler.py. Github Repository. https://github.com/shubhomoydas/ad_examples/blob/master/ad_examples/ad/spectral_outlier.py

[38] Kashef, R., Gencarelli, M. and Ibrahim, A. (2020) Classification of Outlier's Detection Methods Based on Quantitative or Semantic Learning. In: Fadlullah, Z. and Khan Pathan, A.S., Eds., *Combating Security Challenges in the Age of Big Data. Advanced Sciences and Technologies for Security Applications*, Springer, Cham, 45-59. https://doi.org/10.1007/978-3-030-35642-2_3

[39] Pedregosa, *et al.* (2011) Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, **12**, 2825-2830. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.cluster_optics_dbscan.html

[40] Polygon.io. (2020) Polygon. Io-Real-Time Stock Apis, Forex and Crypto. https://polygon.io/

[41] Dautov, Ç.P. and Özerdem, M.S. (2018) Wavelet Transform and Signal Denoising Using Wavelet Method. 2018 26*th Signal Processing and Communications Applications Conference* (*SIU*), Izmir, 2-5 May 2018, 1-4. https://doi.org/10.1109/SIU.2018.8404418

[42] Ding, Y. and Selesnick, I.W. (2015) Artifact-Free Wavelet Denoising: Non-Convex Sparse Regularization, Convex Optimization. *IEEE Signal Processing Letters*, **22**, 1364-1368. https://doi.org/10.1109/LSP.2015.2406314

[43] MProx (2019) Wavelet-Denoising. Github Repository. https://github.com/MProx/Wavelet-denoising/blob/master/wavelets.py