sensors

Article

# Signal Processing for Time Domain Wavelengths of Ultra-Weak FBGs Array in Perimeter Security Monitoring Based on Spark Streaming

**Zhenhao Yu [1,2], Fang Liu [1,2,*], Yinquan Yuan [1], Sihan Li [2] and Zhengying Li [1,3]**

1 National Engineering Laboratory for Fiber Optic Sensing Technology, Wuhan University of Technology, Wuhan 430070, China; dwyanesir_yu@whut.edu.cn (Z.Y.); ymyyq@whut.edu.cn (Y.Y.); zhyli@whut.edu.cn (Z.L.)
2 School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070, China; lsh199395@whut.edu.cn
3 School of Information Engineering, Wuhan University of Technology, Wuhan 430070, China
* Correspondence: fangliu@whut.edu.cn; Tel.: +86-27-87651850-8602

Received: 27 July 2018; Accepted: 1 September 2018; Published: 4 September 2018

**Abstract:** To detect perimeter intrusion accurately and quickly, a stream computing technology was used to improve real-time data processing in perimeter intrusion detection systems. Based on the traditional density-based spatial clustering of applications with noise (T-DBSCAN) algorithm, which depends on manual adjustments of neighborhood parameters, an adaptive parameters DBSCAN (AP-DBSCAN) method that can achieve unsupervised calculations was proposed. The proposed AP-DBSCAN method was implemented on a Spark Streaming platform to deal with the problems of data stream collection and real-time analysis, as well as judging and identifying the different types of intrusion. A number of sensing and processing experiments were finished and the experimental data indicated that the proposed AP-DBSCAN method on the Spark Streaming platform exhibited a fine calibration capacity for the adaptive parameters and the same accuracy as the T-DBSCAN method without the artificial setting of neighborhood parameters, in addition to achieving good performances in the perimeter intrusion detection systems.

**Keywords:** FBGs signal processing; perimeter security monitoring; spark streaming; AP-DBSCAN

## 1. Introduction

With the widespread technological development of society, security issues have become increasingly prominent. Thanks to the recent progress of modern science and technology, better solutions have become available to solve security problems. Among them are fiber Bragg grating interference technology [1–3], big data processing technology [4–6], machine learning [7,8] and stream processing technology [9–11]. Recently, our team has realized the online writing of an ultra-weak FBG (UWFBG) array during the drawing process of single mode fibers (SMF). A large-scale UWFBG array is made up of hundreds or thousands of identical-wavelength FBGs with a reflectivity of about $-50$ dB for each FBG. Such a large-scale UWFBG sensor array has attracted a great deal of attention in major engineering monitoring, because of its low cost, low crosstalk, and strong multiplexing capacity [12–14]. In particular, the UWFBGs have the advantages of small size, favorable wavelength selectivity, and anti-electromagnetic interference, and so they are widely used in perimeter security and structural health monitoring.

The physical parameters of these UWFBGs, such as their reflected powers and Bragg wavelengths, vary with external vibration signals. The external signals are extracted from the light signs through demodulation and further data processing. As the demodulated data shows characteristics of having a large capacity, much noise, and a high frequency, it is necessary to detect abnormal data affected by intrusion in a real-time and more accurate way. The outliers can be analyzed to find more abnormal classes by using a clustering algorithm. Thus, how to improve the capability of existing machine learning algorithms to process large-scale data in real-time has become a hot issue.

Handling large-scale data streams requires the support of stream computing [15]. Stream data is a type of data form in the big data environment that was born at the end of 20th century and has gradually become a hot issue in the development of cloud computing and the Internet of Things. Spark Streaming [16] is an extension to the core spark API (application programming interface) [17] that can handle real-time data streams by DStream (discretized Stream). The DStream is expressed by a continuous set of RDDs (resilient distributed datasets) over a time series. Each RDD contains a data stream at a particular time interval. Input data, which is taken from Spark Streaming according to the batch size (such as 1 s), is broken down into segments; each segment is converted to RDD and the results of the RDD operation are stored in the memory. Therefore, Spark Streaming can be used to process large-scale optical fiber grating stream data.

The stream data of perimeter intrusion contains a large amount of information, and the feature parameters of the data are extracted for cluster analysis [18–20]. Based on the characteristics of the event types, different clusters can be formed whenever a perimeter intrusion event occurs, which forms a new cluster. More focus should be put on this new type of event in order to detect anomalies in stream data. In the face of unknown distribution data, such as the data collected in perimeter security, the clustering method takes advantage of the relationships among the data objects to gather data in different classes, which is actually an unsupervised way of finding the optimal partition.

The traditional density-based spatial clustering of applications with noise (T-DBSCAN) [21] was proposed by Martin Ester et al. This T-DBSCAN method depends on the choice and calibration of two neighborhood parameters, namely the characteristic size of clusters ($\varepsilon$) and the minimum number of points in a cluster ($N_{\min}$), and so the choice of neighborhood parameters has a great influence on the determination of clusters and it will enable the calibration of a large workload of neighborhood parameters. In recent years, some scholars have tried to improve the T-DBSCAN algorithm through clustering analysis. For example, Li et al., designed a modified DBSCAN to identify fixations in eye-tracking data, thus including the advantages of the classical fixation identification method [22]. Edla et al. proposed a prototype-based modified DBSCAN algorithm to cluster the gene expression data and speed up the DBSCAN algorithm [23]. Cai et al., proposed an improved DBSCAN algorithm that is insensitive to input parameters by adding the connection information of the clusters and merging the related clusters [24], but this method is incapable of selecting neighborhood parameters automatically to achieve unsupervised anomaly detection. Feng et al., proposed an adaptive DBSCAN algorithm for constellation reconstruction and modulation identification [25] by choosing the parameter $N_{\min}$, and then different values of $\varepsilon$ were introduced to the trial clustering, the optimal clustering was obtained by evaluating the validity of each cluster. However, it is hard to define the parameter $N_{\min}$ in the case of trial clustering.

In this paper, an adaptive-parameters DBSCAN (AP-DBSCAN) method was proposed, in which two neighborhood parameters, $\varepsilon$ and $N_{\min}$, were determined automatically by time-domain statistical analysis. Then, AP-DBSCAN was implemented on Spark Streaming to deal with the problems of data stream collection and real-time analysis. After that, a perimeter intrusion detection system based on suspended UWFBG sensing arrays and UWFBG sensing arrays that were buried under the ground was constructed. Finally, a number of sensing and processing experiments were finished and analyzed.

## 2. Materials and Methods

### 2.1. T-DBSCAN

DBSCAN is a famous density based clustering technique [21]. A cluster in this model is described as a linked region that exceeds a given density threshold. The functioning of DBSCAN is directed by two definitions, namely density-reachability and density-connectability, which depend on two predefined parameter values: the size of the neighborhood, denoted by $\varepsilon$, and the number of neighborhood points in a cluster, denoted by $N_{\min}$. In T-DBSCAN, one begins with a random point x and it finds all of the points that are density-reachable from x with respect to $\varepsilon$ and $N_{\min}$. It is obvious that no points are density-reachable from x when x is a border point; in this case, the T-DBSCAN begins with an unclassified point to repeat the same process, and so the two predefined parameters $\varepsilon$ and $N_{\min}$ decide the quality and efficiency of clusters.

### *2.2. AP-DBSCAN Algorithm*

In the T-DBSCAN method, the values of $\varepsilon$ and $N_{\min}$ are regulated by the users. To avoid human intervention, we proposed an unsupervised clustering method. A sample set composed of $n$ sensing signals, $\text{Signs}_n = \{(x_l), l = 1, 2, \ldots, n\}$, with the sampling frequency ($f$) and a quantity of $n_f = n/f$, can be denoted by

$$\text{Signs}_{\,n} = \{(\text{s}_m), m = 0, 1, 2, \ldots, n_\text{f}\} \tag{1}$$

where

$$\text{s}_m = \left\{x_{mf+1}\,, x_{mf+2}, \ldots, x_{mf+f}\right\} \tag{2}$$

Then, a set of characteristic parameters, energy and average amplitude, can be calculated, as follows:

$$energy = \left\{\sum_{i=1}^{f} x_{mf+i}^2, m = 0, 1, 2, \ldots, n_\text{f}\right\} \tag{3}$$

$$average\ amplitude = \left\{\frac{1}{f}\sum_{i=1}^{f}\left|x_{mf+i}\right|, m = 0, 1, 2, \ldots, n_\text{f}\right\} \tag{4}$$

A set of sample characteristic parameters can be described as:

$$\text{T} = \left\{\left(\sum_{i=1}^{f} x_i^2, \frac{1}{f}\sum_{i=1}^{f}|x_i|\right), \left(\sum_{i=1}^{f} x_{i+f}^2, \frac{1}{f}\sum_{i=1}^{f}\left|x_{i+f}\right|\right), \left(\sum_{i=1}^{f} x_{i+2f}^2, \frac{1}{f}\sum_{i=1}^{f}\left|x_{i+2f}\right|\right), \ldots, \left(\sum_{i=1}^{f} x_{i+n_\text{f}f}^2, \frac{1}{f}\sum_{i=1}^{f}\left|x_{i+n_\text{f}f}\right|\right)\right\} \tag{5}$$

These characteristic data contain different dimensions. An effective normalization is needed to eliminate the influence of target dimensions, and so the min-max normalization method was used, which enables the mapping of energy and average amplitude in the range of $[0, 1]$ by the linear transformation of the characteristic set T. Let the horizontal axis and vertical axis in the range of $[0, 1]$ denote normalized energy and normalized average amplitude, respectively; then, the original one-dimensional data were converted into two-dimensional normalized data. Then, a symmetric distance matrix which describes the distances between all pairs of points may be constructed, as follows:

$$\mathbf{D}_\text{t} = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1t} \\ d_{21} & d_{22} & \cdots & d_{2t} \\ \vdots & \vdots & \ddots & \vdots \\ d_{t1} & d_{t2} & \cdots & d_{tt} \end{bmatrix} \tag{6}$$

where $t = 1 + n/f$ is the number of the characteristic sample sets and $d_{ij}$ is the distance between points $i$ and $j$.

Sorting the elements of each row in the matrix $\mathbf{D_t}$ from small to large in turn, a new matrix $\mathbf{D_s}$ can be obtained. In the matrix $\mathbf{D_s}$, all of the elements at the first column are zero, and the elements at the kth column (k > 1) are the $(\text{k} - 1)^{\text{th}}$ closer distances. The sorted matrix $\mathbf{D_s}$ can be represented by column matrices:

$$\mathbf{D}_\text{s} = (\zeta_1, \zeta_2, \ldots, \zeta_t\ ), \quad \zeta_i = (d_{1i}, d_{2i}, \ldots, d_{ti})^\text{T} \tag{7}$$

For all the column matrices, calculating their $J$ values gives the following:

$$J_i = J(d_{1i}\ , d_{2i}, \ldots, d_{ti}) = \frac{1}{2}\left(\sum_{i=1}^{t} d_{1i} - \frac{1}{t}\sum_{i=1}^{t} d_{1i}\right)^2 \ i = 1, 2, \ldots, t \tag{8}$$

Then, to find the characteristic column matrix, which produces a minimum of all $J$'s values, this process can be denoted by:

$$\gamma = \mathrm{argmin}(J(\zeta_i\ ), i = 1, 2, \ldots, t) \tag{9}$$

Thus, the characteristic column matrix $\gamma = \zeta_{imin} = (d_{1,imin}, d_{2,imin}, \ldots, d_{t,imin})^{\mathrm{T}}$ can be obtained. Progressively, the maximum distance in the characteristic column matrix was assigned as $\varepsilon$.

After the determination of $\varepsilon$, and then performing an arithmetic mean for the number of points within the $\varepsilon$-neighborhood in the entire data set, an optimal value of the point number in each cluster can be obtained:

$$N_{\min} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{10}$$

where $X_i$ is the number of points in the $\varepsilon$-neighborhood of each point.

## 2.3. *AP-DBSCAN on Spark Streaming*

Figure 1 describes the implementation process of the proposed AP-DBSCAN method on Spark Streaming, which decomposes streaming computing into a series of short batch jobs. The batch engine is Spark Core, which divides the input data into pieces of data according to the batch size (for example, 4 s). The data were converted to the RDD in Spark, and then the transformation operation was changed to the RDD transformation operation, and each RDD is the data conversion of $T$ s demodulated by the fiber grating signal processor.

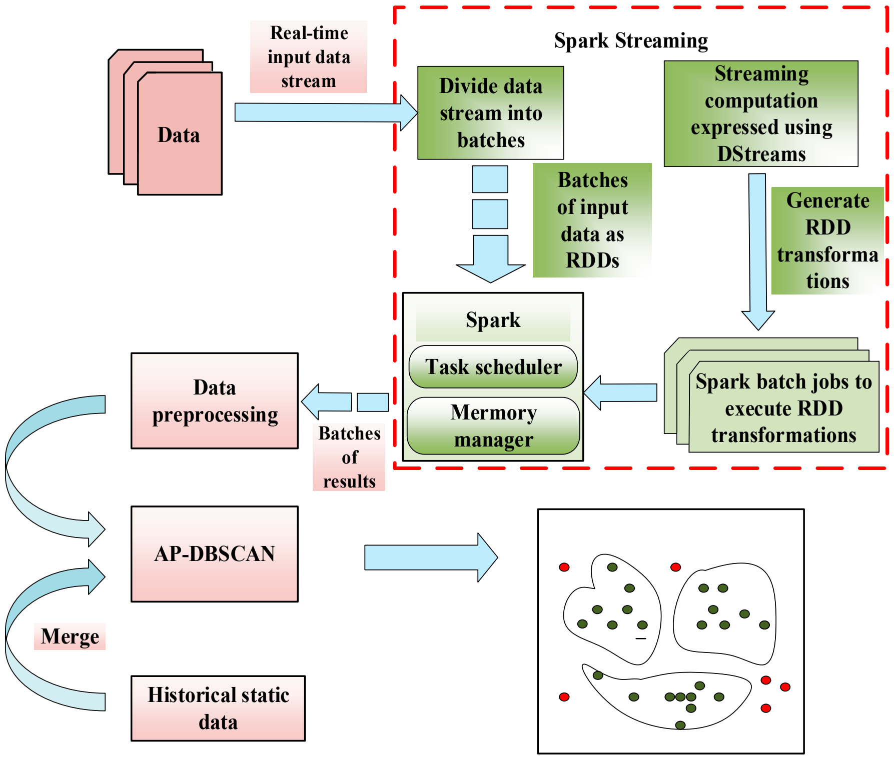
Data
Real-time input data stream
Spark Streaming
Divide data stream into batches
Streaming computation expressed using DStreams
Batches of input data as RDDs
Generate RDD transformations
Spark
Task scheduler
Mermory manager
Spark batch jobs to execute RDD transformations
Data preprocessing
Batches of results
AP-DBSCAN
Merge
Historical static data

**Figure 1.** Adaptive parameters density-based spatial clustering of applications with noise (AP-DBSCAN) clustering analysis based on the Spark Streaming mechanism.

The main steps in AP-DBSCAN on Spark Streaming are shown in Algorithm 1, and the workflows of the algorithm are shown in Figure 2. Box-plots [26] and fast Fourier transform [27] are employed to deal with noises. The feature sets of normal data in the first RDD, which mean that there is no intrusion, are obtained by Equations (3) and (4), and are then mixed to the proposed AP-DBSCAN in case the abnormal data appears at the beginning. The noise data is reconstructed and added to the normal data from AP-DBSCAN. On the DStream, each piece of data is continuously updated by RDD. If there is no abnormal data in a certain RDD, the output of the RDD includes normal feature samples of the last RDD and the current RDD; if there is abnormal data in a certain RDD, then the output of the RDD includes not only the normal feature samples of the last RDD and the current RDD, but also the abnormal samples of the current RDD. Thus, the clustering result of each RDD is achieved through AP-DBSCAN when the distinguished abnormal samples are output and the normal data samples are mixed with the following RDD.

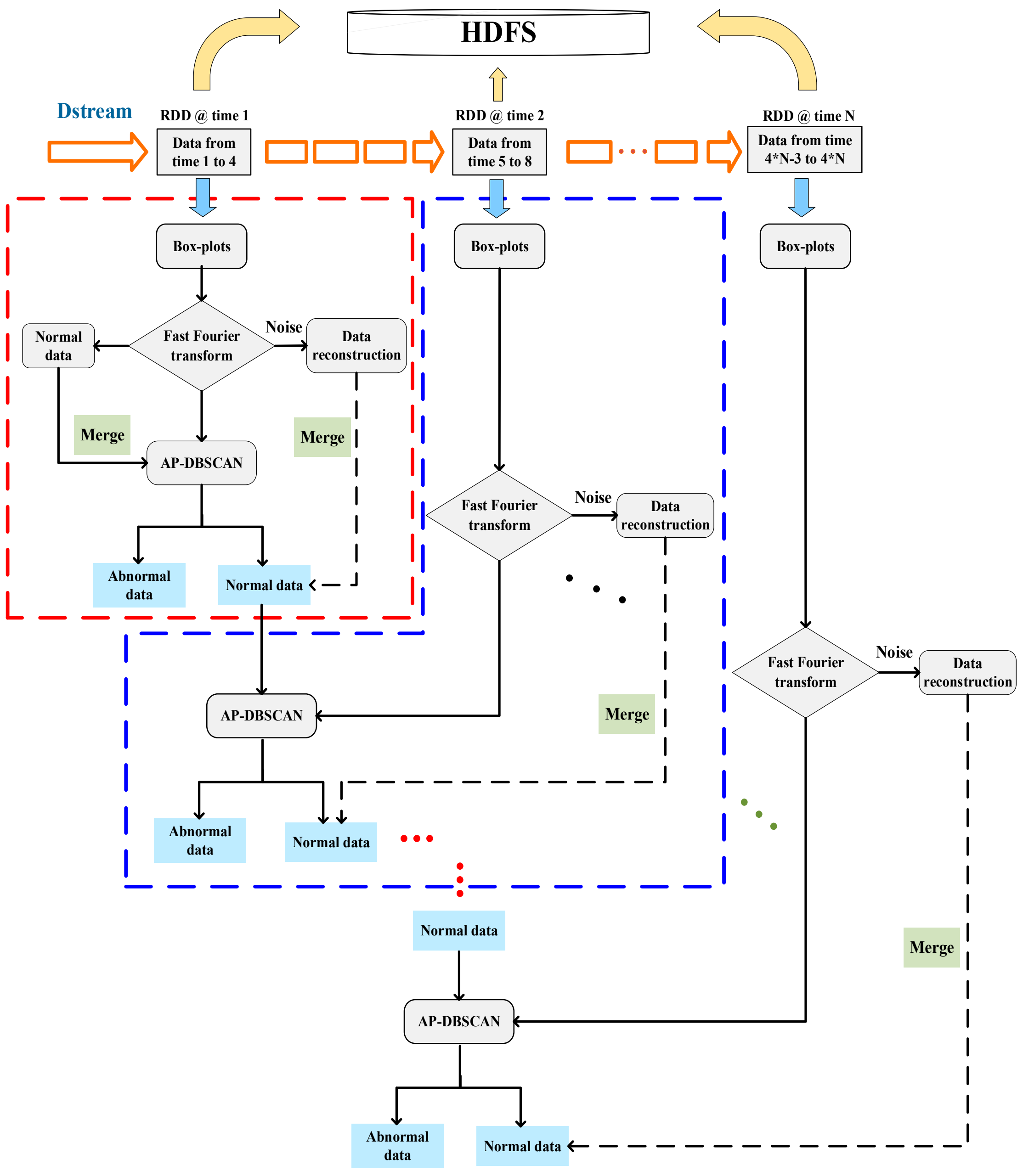

**Figure 2.** The workflows of AP-DBSCAN implementation on Spark Streaming.

**Algorithm 1.** Main steps in AP-DBSCAN on Spark Streaming.

```
1: Input:
        The training sets of $n$ workers: $D_n = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
Normal data: $N_t = \{(x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t)\}$
2: Step1: Create a local streaming context with two working thread and a batch interval of 4 s.
3: Step2: Create an input in DStream.
4: Step3: Operate DStream:
        Convert segment data and normal data to RDD, perform the first AP-DBSCAN to get the result of the
clustering:
                $D_i : \{(x_1, y_1), (x_2, y_2), \ldots, (x_i, y_i)\} \cup N_t : \{(x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t)\}$
                  $\rightarrow$first RDD$\rightarrow$AP-DBSCAN$\rightarrow$the first clustering result
        While input DStream = true
        Abnormal data is separated from the first result, normal data is retained and mixed into the next data;
        Perform AP-DBSCAN to get the result of clustering.
5: Step4: Start Spark Streaming.
6: Output: The results of clustering on each RDD.
```

## 3. Results and Analysis

### 3.1. Monitoring System Based on the UWFBG Array

The architecture of the intrusion monitoring and identification system is shown in Figure 3. The sensing system is composed of a quasi-distribution UWFBG array that was prepared on a drawing single mode silica optical fiber, a 1550-nm laser source (RIO, narrow frequency laser module, 1 kHz of line width), an FBG signal processor, a detector (4-way photoelectric detection plate, self-control, bandwidth is 60 MHz), and a computer. UWFBGs with the same Bragg wavelengths were used as a string of vibration detectors to encapsulate the external vibration signals near the optical fibers. In our experiments, two kinds of UWFBG sensing arrays were prepared at a distance of every 5 m: one is the suspended UWFBG sensing array with a length of 100 m (20 sensors), which was fixed along a railing; the other is the buried UWFBG sensing array with a length of 300 m (60 sensors), which was buried under the ground.

All of the wavelength shift signals from the UWFBGs were transmitted to the signal processor based on Mach-Zehnder interference (MZI). The grating signal processor was connected to a computer through a network line, sending and receiving the data by the user datagram protocol (UDP). The computer and the software received the data stream from the grating signal processor, pushed the stream data to Spark Streaming for real-time processing, and saved the data to the Hadoop distributed file system (HDFS). As abnormal data appeared, the computer output them in real-time, and finally actualized the intelligent analysis and pattern recognition of the intrusion signals.

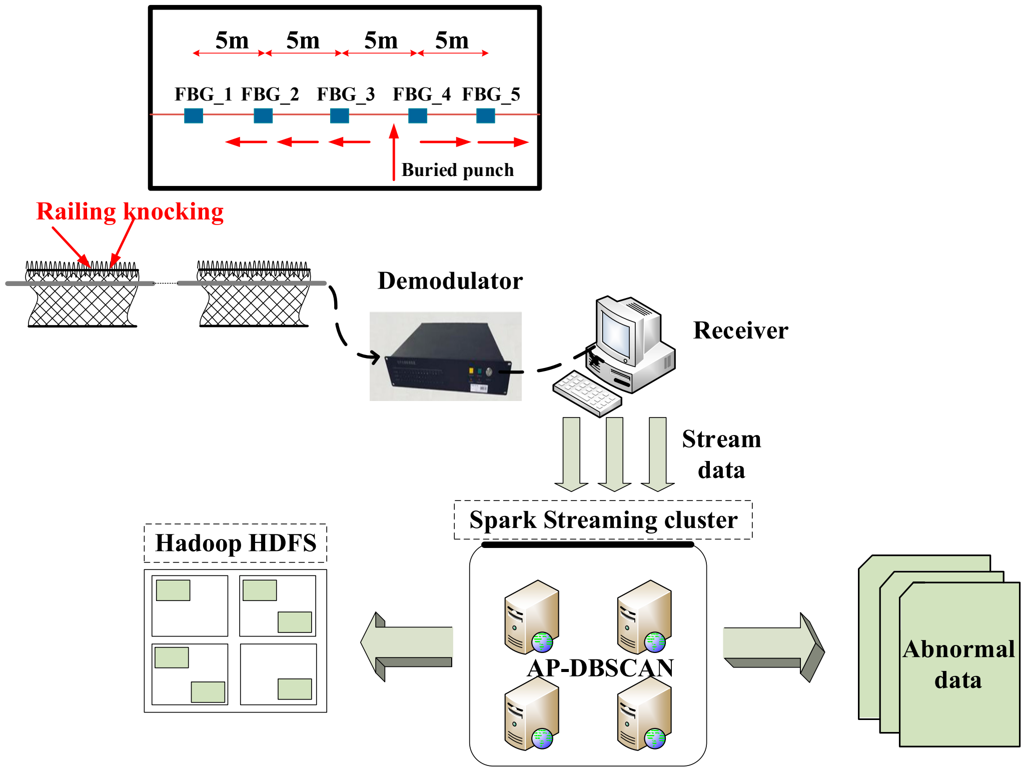

**Figure 3.** Architecture of the intrusion monitoring and identification system.

The data acquisition methods are as follows: the continuous light from an amplified spontaneous emission (ASE) was modulated into a nanosecond pulse by a semiconductor optical amplifier. The pulse light was launched into the fiber with uniformly distributed UWFBGs by a circulator, and then a pulse train could be realized. A phase demodulation unit consisting of an unbalanced MZI, a 3 $\times$ 3 coupler and three detectors was used to restore the vibration signal. The unbalanced paths of the MZI separated each reflected pulse to two pulses; the slower pulse from the closer UWFBGs coincided with the faster pulse from the further UWFBGs, and the coherence would be maintained. Phase perturbations that are caused by vibrations between the two adjacent UWFBGs can be demodulated from the interference light pulse. According to optical time domain reflectometry, the correspondence relationships between the interference light pulse and sensing position were established.

The experiments were finished on a cluster with four nodes: a master node and three compute nodes. The configuration of each node is as follow: 3.4 GHz Intel Core i7-6700 processor, 8 M cache, 4 G memory, and 1 TB storage. The software used are Spark 1.6.1 and the Ubuntu 16.04 operating system, and the sampling frequency is 100 Hz.

## *3.2. Signal Processing for Railing Sensors*

Three kinds of railing intrusion behaviors, namely knock, shake, and climb, were simulated, and the corresponding vibration signals are shown in Figure 4a–c. The DBCSAN and AP-DBSCAN calculations were finished by selecting the data of three behaviors and merging static data, respectively. The corresponding clustering effects are shown in Figure 5, where the horizontal axis and vertical axis denote the normalized energy and the normalized average amplitude, respectively. According to the distance between the mass center of each cluster and the origin, one can judge the type of behaviors and each cluster represents a behavior.

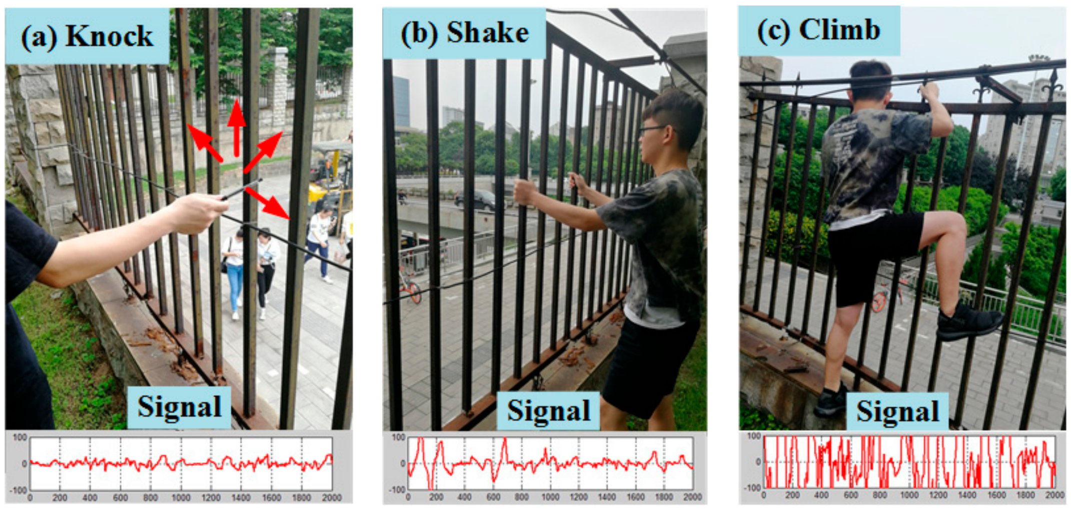

**Figure 4.** Three types of railing intrusion. (**a**) Knocking the railing; (**b**) shaking the railing; and, (**c**) climbing on the railing.

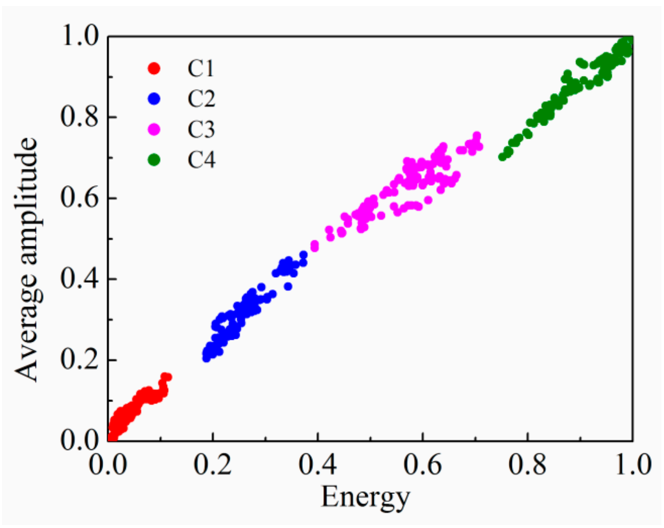

**Figure 5.** AP-DBSCAN results of three kinds of railing intrusion behaviors.

In addition, the number of points in each cluster and the number of clusters corresponding to all the behaviors given by the T-DBSCAN and AP-DBSCAN methods were calculated and are shown in Table 1. The number of data for static, knocking, shaking, and climbing the rail are represented by C1, C2, C3, and C4, respectively; the results showed that the AP-DBSCAN method can achieve the same precision as T-DBSCAN without setting neighborhood parameters manually, being conductive to the realization of automatic detection in the perimeter intrusion detection systems.

**Table 1.** Comparison of calculated data by two methods for railing sensors.

| **Data Set** | **Clustering Algorithm** | |
|---|---|---|
| | **T-DBSCAN** | **AP-DBSCAN** |
| C1 | 119 | 119 |
| C2 | 113 | 113 |
| C3 | 111 | 111 |
| C4 | 121 | 121 |
| Number of clusters | 4 | 4 |

Finally, according to the methods by K-means [28], FCM (Fuzzy C-means) [29], and AP-DBSCAN, the misclassified patterns, the computation time and the error rate (ER) were computed and compared; their data are shown in Table 2, where the error rate was computed, as follows:

$$ER = \frac{Number\ of\ misclassified\ objects}{Total\ number\ of\ objects} \times 100\% \tag{11}$$

Smaller ERs given by AP-DBSCAN indicated the AP-DBSCAN method can produce better results than the other methods.

**Table 2.** Comparisons of misclassified patterns, computation time and the error rate (ER) for different sizes of data sets for railing sensors. A: K-means; B: Fuzzy C-means (FCM); C: AP-DBSCAN.

| Data Size (kB) | Misclassified Patterns (kB) | | | Computation Time (s) | | | ER (%) | | |
|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | A | B | C | A | B | C |
| 150 | 12 | 3 | **3** | 10 | 10 | **11** | 8.0 | 2.0 | **2.0** |
| 185 | 15 | 4 | **5** | 27 | 29 | **25** | 8.1 | 2.1 | **3.0** |
| 286 | 26 | 10 | **9** | 43 | 44 | **41** | 9.1 | 3.5 | **3.1** |
| 768 | 59 | 24 | **21** | 391 | 401 | **379** | 7.7 | 3.1 | **2.8** |
| 1024 | 93 | 32 | **33** | 578 | 593 | **533** | 9.1 | 3.2 | **3.3** |
| 1625 | 131 | 49 | **47** | 1601 | 1701 | **1567** | 8.1 | 3.0 | **3.0** |

### *3.3. Signal Processing for Buried Sensors*

The underground sensing optical cable was buried under the ground at a depth of half a meter so that the optical cables were less affected by noise. We simulated five behaviors that influence the underground cable: walking on the buried cable, walking parallel to the cable at distances of 20, 40, 60 cm from the cable, and static standing, respectively. Each behavior records data for 20 s, and the corresponding vibration signals are shown in Figure 6a–d. Due to low noise, the collected data is processed by difference denoising methods, and then the AP-DBSCAN method is used to calculate the clustering based on the data of five behaviors. The corresponding clustering effects are shown in Figure 7, where the horizontal axis and vertical axis denote the normalized energy and the normalized average amplitude, respectively. The AP-DBSCAN on Spark Streaming and that on a single machine are compared with the clustering speed at an interval of 4 s. The abnormal behavior is obtained by clustering according to the data characteristics of different behaviors. Each cluster represents one kind of behavior; the abscissa and the ordinate of point denote energy and frequency amplitude, respectively. According to the distance from the origin to the center of the cluster, one can determine the type of the behaviors.

Then, we calculate the number of clusters that form each behavior and the number of points in each cluster, as given by T-DBSCAN and AP-DBSCAN. The number of data sets of static standing, walking parallel to the cable at distances of 60, 40, and 20 cm from the cable, and walking on the buried cable is denoted by C1, C2, C3, C4, and C5, respectively. According to the same sample data, the experimental results of the two methods are shown in Table 3, which exhibits that the AP-DBSCAN method can achieve the same accuracy as the T-DBSCAN without artificial setting of neighborhood parameters; thus, the AP-DBSCAN method can save a large amount of labor time.

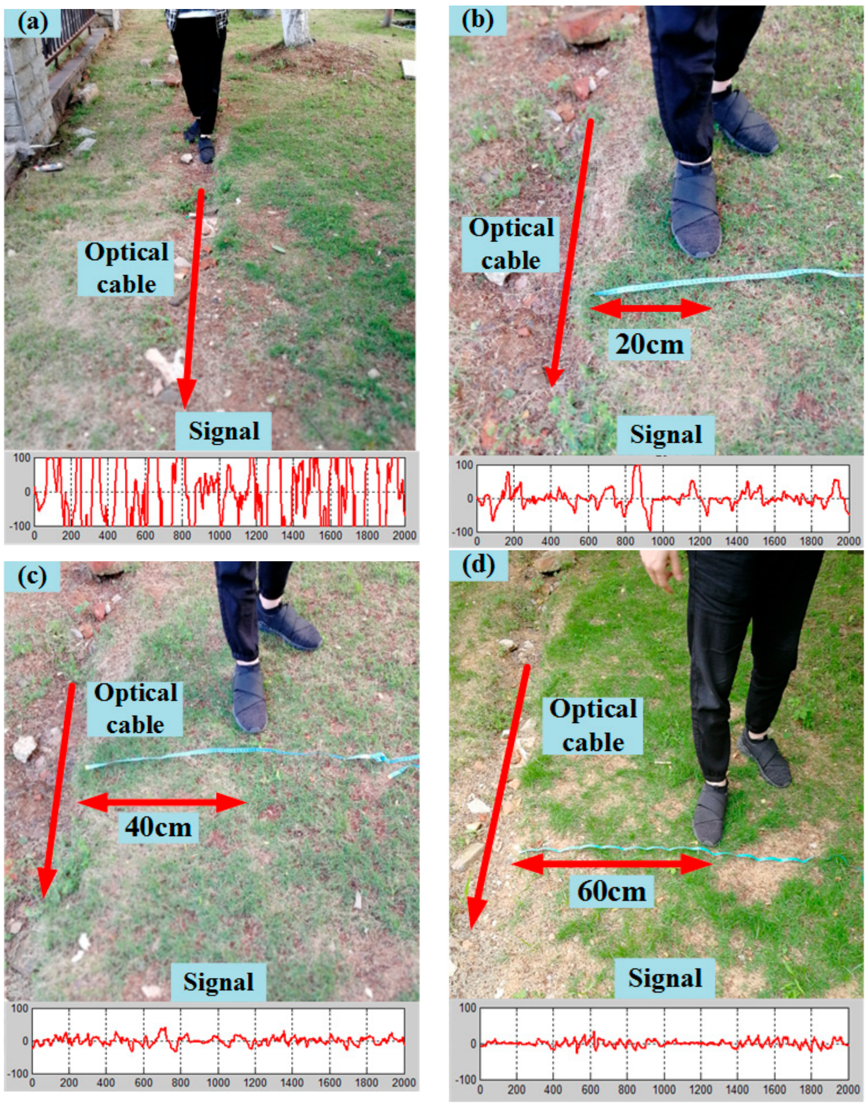

**Figure 6.** Four types of underground fence intrusion. (**a**) Walking on the buried cable; (**b**) walking parallel to the cable at a distance of 20 cm; (**c**) walking parallel to the cable at a distance of 40 cm; and, (**d**) walking parallel to the cable at a distance of 60 cm.

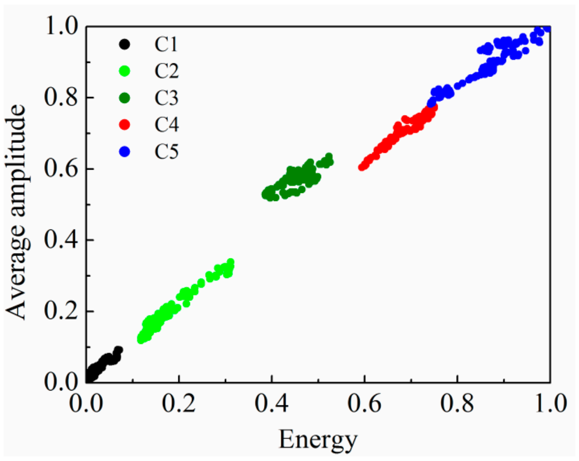

**Figure 7.** AP-DBSCAN results of four kinds of buried intrusion behaviors.

**Table 3.** Comparison of calculated data by two methods for buried sensors.

| **Data Set** | **Clustering Algorithm** | |
|---|---|---|
| | **T-DBSCAN** | **AP-DBSCAN** |
| C1 | 116 | 116 |
| C2 | 115 | 115 |
| C3 | 115 | 115 |
| C4 | 115 | 115 |
| C5 | 114 | 114 |
| Number of clusters | 5 | 5 |

Thirdly, according to the methods by K-means [28], FCM (Fuzzy C-means) [29], and AP-DBSCAN, the misclassified patterns, the computation time, and the error rate (ER) were computed and compared; their data are shown in Table 4. One can see that the smaller ERs given by AP-DBSCAN indicated that the AP-DBSCAN method can produce better results than the other methods.

**Table 4.** Comparisons of misclassified patterns, computation time and the error rate (ER) for different sizes of data sets for buried sensors. A: K-means; B: FCM; C: AP-DBSCAN.

| **Data Size (kB)** | **Misclassified Patterns (kB)** | | | **Computation Time (s)** | | | **ER (%)** | | |
|---|---|---|---|---|---|---|---|---|---|
| | **A** | **B** | **C** | **A** | **B** | **C** | **A** | **B** | **C** |
| 131 | 10 | 5 | **2** | 9 | 9 | **10** | 7.6 | 3.8 | **1.5** |
| 254 | 23 | 15 | **7** | 40 | 42 | **37** | 9.1 | 5.9 | **2.8** |
| 552 | 46 | 27 | **13** | 287 | 301 | **266** | 8.3 | 4.9 | **2.4** |
| 783 | 62 | 34 | **23** | 399 | 420 | **391** | 7.9 | 4.3 | **2.9** |
| 1131 | 101 | 81 | **39** | 583 | 606 | **542** | 8.9 | 7.1 | **3.4** |
| 1721 | 140 | 121 | **51** | 1721 | 1835 | **1643** | 8.1 | 7.0 | **3.0** |

Finally, the time response of AP-DBSCAN on Spark Streaming was investigated also. In order to test the timeliness of the algorithm, several invasion scenarios were simulated. At a distance of 1 m from one side of the detection optical cable, one person walked to the optical cable vertically and arrived at the ground over the optical cable, and then walked 1 m distance to other side of the optical cable. Meanwhile, 100 pieces of data are recorded every 1 s and calculated every 2 s; the clustering data and the number of abnormal events are shown in Figure 8. It can be seen that the data can be received and processed in an effective time. To test the performance of the AP-DBSCAN method on Spark Streaming, the time responses was measured and shown in Figure 9. There is a small difference between AP-DBSCAN on a single machine and AP-DBSCAN on Spark Streaming when the test data is small. However, when the test data is very big, the response time by the AP-DBSCAN on Spark Streaming is significantly superior to the response time by AP-DBSCAN on the single machine.

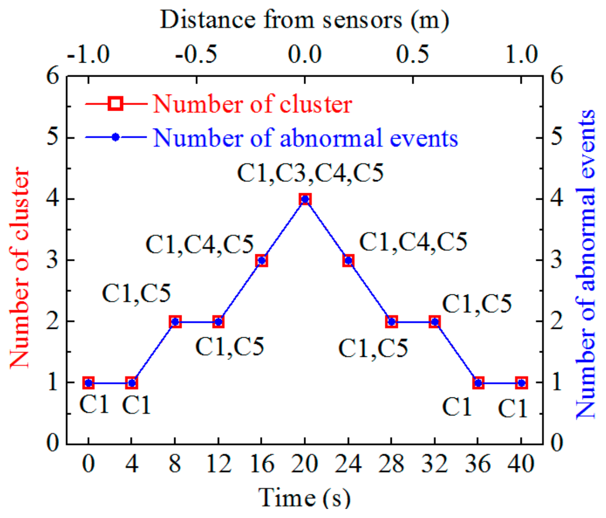

**Figure 8.** Time response of AP-DBSCAN on Spark Streaming.

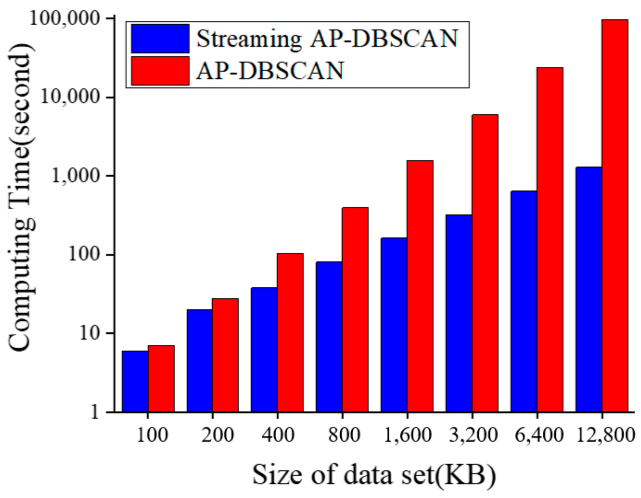

**Figure 9.** Computing times by AP-DBSCAN and AP-DBSCAN on Spark Streaming.

## 4. Conclusions

In this paper, we propose the AP-DBSCAN algorithm with adaptive parameters on the Spark Streaming platform, solving the problem of the real-time anomaly detection of large-scale data in perimeter security. The preprocessing of the algorithm combines the Box-plots and the fast Fourier transform, and it is necessary to make certain that the data stream of a segment is mixed with the normal data stream of the previous segment to detect the abnormal data of different types. In the verification experiment of AP-DBSCAN, the proposed algorithm improves the unsupervised capability of T-DBSCAN and can detect abnormal conditions of large-scale data in real-time, providing better convenience and service for perimeter security.

**Author Contributions:** Conceptualization, F.L.; Methodology, F.L. and Z.Y.; Software, Z.Y.; Validation, Z.Y. and S.L.; Writing-Original Draft Preparation, Z.Y.; Formal Analysis and Writing-Review & Editing, Y.Y.; Project Administration and Funding Acquisition, Z.L.

**Funding:** This research was funded by the National Natural Science Foundation of China grant numbers 61735013 and 61402345, and the Natural Science Foundation of Hubei Province of China grant number 2018CFA056.

**Acknowledgments:** This work is supported by National Natural Science Foundation of China and Natural Science Foundation of Hubei Province of China.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Yu, H.H.; Zheng, Y.; Guo, H.Y. Research progress in online preparation techniques of fiber Bragg gratings on optical fiber drawing tower. *J. Funct. Mater.* **2014**, *45*, 12001–12005.
2. Allwood, G.; Hinckley, S.; Wild, G. Optical Fiber Bragg grating based intrusion detection systems for homeland security. In Proceedings of the IEEE Sensors Applications Symposium (SAS), Galveston, TX, USA, 19–21 February 2013; pp. 66–70.
3. Catalano, A.; Bruno, F.A.; Pisco, M. An Intrusion Detection System for the Protection of Railway Assets Using Fiber Bragg Grating Sensors. *Sensors* **2014**, *14*, 18268–18285. [CrossRef] [PubMed]
4. Zaharia, M.; Xin, R.S.; Wendell, P. Apache spark: A unified engine for big data processing. *Commun. ACM* **2016**, *59*, 56–65. [CrossRef]
5. Stai, E.; Karyotis, V.; Papavassiliou, S. A Hyperbolic Space Analytics Framework for Big Network Data and their Applications. *IEEE Netw.* **2016**, *30*, 11–17. [CrossRef]
6. Jia, Z.; Xue, C.; Chen, G.; Zhan, J.; Zhang, L.; Lin, Y.; Hofstee, P. Auto-tuning Spark big data workloads on POWER8: Prediction-based dynamic SMT threading. In Proceedings of the 2016 International Conference on Parallel Architectures and Compilation, Haifa, Israel, 11–15 September 2016; pp. 387–400.
7. Robert, C. Machine learning, a probabilistic perspective. *Chance* **2014**, *27*, 62–63. [CrossRef]
8. Snoek, J.; Larochelle, H.; Adams, R.P. Practical Bayesian optimization of machine learning algorithms. In Proceedings of the International Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, 3–6 December 2012; Curran Associates Inc.: Red Hook, NY, USA, 2012; pp. 2951–2959.
9. O'Callaghan, L.; Mishra, N.; Meyerson, A.; Guha, S. Streaming-data algorithms for high-quality clustering. In Proceedings of the IEEE International Conference on Data Engineering, San Jose, CA, USA, 26 February–1 March 2002; pp. 685–694.
10. Fenandez-Rodriguez, J.Y.; Alvarez-Garcia, J.A.; Fisteus, J.A.; Luaces, M.R.; Magana, V.C. Benchmarking real-time vehicle data streaming models for a Smart City. *Inf. Syst.* **2017**, *72*, 62–76. [CrossRef]
11. Lee, K.; Lee, C.G. Lazy Approaches for Interval Timing Correlation of Sensor Data Streams. *Sensors* **2010**, *10*, 5329–5345. [CrossRef] [PubMed]
12. Guo, H.; Liu, F.; Yuan, Y.; Yu, H.; Yang, M. Ultra-weak FBG and its refractive index distribution in the drawing optical fiber. *Opt. Express* **2015**, *23*, 1094–4087. [CrossRef] [PubMed]
13. Wang, Y.; Gong, J.; Dong, B.; Wang, D.; Shilig, T.J.; Wang, A. A large serial time-division multiplexed fiber Bragg grating sensor network. *J. Lightw. Technol.* **2012**, *40*, 2751–2756. [CrossRef]
14. Zhang, M.L.; Sun, Q.Z.; Wang, Z.; Li, X.; Liu, H.; Liu, D. A large capacity sensing network with identical weak fiber Bragg gratings multiplexing. *Opt. Commun.* **2012**, *285*, 3082–3087. [CrossRef]
15. Zaharia, M.; Das, T.; Li, H. Discretized streams: Fault-tolerant streaming computation at scale. In Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, Farmington, PL, USA, 3–6 November 2013; pp. 423–438.
16. Ghesmoune, M.; Lebbah, M.; Azzag, H. Micro-batching growing neural gas for clustering data streams using spark streaming. *Procedia Comput. Sci.* **2015**, *53*, 158–166. [CrossRef]
17. Lu, Y.; Cao, B.; Rego, C.; Glover, F. A tabu search based clustering algorithm and its parallel implementation on spark. *Appl. Soft Comput.* **2018**, *63*, 97–109. [CrossRef]
18. Miller, Z.; Dickinson, B.; Deitrick, W.; Hu, W.; Wang, A.H. Twitter spammer detection using data stream clustering. *Inf. Sci.* **2014**, *260*, 64–73. [CrossRef]
19. Yue, S.H.; Li, P.; Guo, J.D. A statistical information based clustering approach in distance space. *J. Zhejiang Univ. Sci.* **2005**, *6*, 71–78. [CrossRef]
20. Peng, C.; Guoyou, S.; Shuang, L.; Jian, Y. An incremental density based spatial clustering of application with noise algorithm based on partition index. *J. Comput. Theor. Nanosci.* **2016**, *13*, 10273–10280. [CrossRef]
21. Cao, F.; Ester, M.; Qian, W.; Zhou, A. Density-based clustering over an evolving data stream with noise. In Proceedings of the 2006 SIAM International Conference on Data Mining, Bethesda, MD, USA, 20–22 April 2006; pp. 328–339.

22. Li, B.; Wang, Q.; Barney, E.; Hart, L.; Wall, C.; Chawarska, K.; de Urabain, I.S.; Smith, T.J.; Shic, F. Modified DBSCAN algorithm on oculomotor fixation identification. In Proceedings of the 2016 Biennial ACM Symposium on Eye Tracking Research & Applications, Charleston, SC, USA, 14–17 March 2016; pp. 337–338.
23. Edla, D.R.; Jana, P.K. A prototype-based modified DBSCAN for gene clustering. *Procedia Technol.* **2012**, *6*, 485–492. [CrossRef]
24. Cai, Y. An improved DBSCAN algorithm which is insensitive to input parameters. *Acta Sci. Nat. Univ. Pekinesis* **2004**, *40*, 480–486.
25. Feng, P.; Ge, L. Adaptive DBSCAN-based algorithm for constellation reconstruction and modulation identification. In Proceedings of the IEEE Asia-Pacific Radio Science Conference, Qingdao, China, 24–27 August 2004; pp. 177–180.
26. Spitzer, M.; Wildenhain, J.; Rappsilber, J.; Tyers, M. BoxPlotR: A web tool for generation of box plots. *Nat. Methods* **2014**, *11*, 121–122. [CrossRef] [PubMed]
27. Haji-Hashemi, H.; Norouzi, P.; Safarnejad, M.R.; Larijani, B.; Habibi, M.M.; Raeisi, H. Sensitive electrochemical immunosensor for citrus bacterial canker disease detection using fast fourier transformation square-wave voltammetry method. *J. Electroanal. Chem.* **2018**, *820*, 111–117. [CrossRef]
28. Khan, S.S.; Ahmad, A. Cluster centre initialization algorithm for K-means clustering. *Pattern Recognit. Lett.* **2004**, *25*, 1293–1302. [CrossRef]
29. Yang, J.; Ke, Y.S.; Wang, M.Z. An adaptive clustering segmentation algorithm based on FCM. *Turk. J. Electr. Eng. Comput. Sci.* **2017**, *25*, 4533–4544. [CrossRef]