A Correlation based Approach to Differentiate between an Event and Noise in Internet of Things

Dina ElMenshawy[1], Waleed Helmy[2]
Information Systems Department, Faculty of Computers and Information
Cairo University
Egypt

***Abstract*—Internet of Things (IoT) is considered a huge enhancement in the field of information technology. IoT is the integration of physical devices which are embedded with electronics, software, sensors, and connectivity that allow them to interact and exchange data. IoT is still in its beginning so it faces a lot of obstacles ranging from data management to security concerns. Regarding data management, sensors generate huge amounts of data that need to be handled efficiently to have successful employment of IoT applications. Detection of data anomalies is a great challenge that faces the IoT environment because, the notion of anomaly in IoT is domain dependent. Also, the IoT environment is susceptible to a high noise rate. Actually, there are two main sources of anomalies, namely: an event and noise. An event refers to a certain incident which occurred at a specific time, whereas noise denotes an error. Both event and noise are considered anomalies as they deviate from the remaining data points, but actually they have two different interpretations. To the best of our knowledge, no research exists addressing the question of how to differentiate between an event and noise in IoT. As a result, in this paper, an algorithm is proposed to differentiate between an event and noise in the IoT environment. At first, anomalies are detected using exponential moving average technique, then the proposed algorithm is applied to differentiate between an event and noise. The algorithm uses the sensors' values and correlation existence between sensors to detect whether the anomaly is an event or noise. Moreover, the algorithm was applied on a real traffic dataset of size 5000 records to evaluate its effectiveness and the experiments showed promising results.**

***Keywords*—Anomaly detection; event; IoT; noise**

I. Introduction

Internet of Things (IoT) is the consolidation of physical objects that are coupled with electronics, software, sensors, and network connectivity, which permit them to capture and transfer data [1]. In IoT, a thing denotes a physical object that contains sensors to interact with the real world through a network to attain specific functions. Things can comprise smart phones, tablets, washing machines, refrigerators, etc. IoT is a network system which connects different communication devices with the internet to establish rapid, reliable, and real time information interchange that assists in intelligent management [2]. The objects capture data about the surrounding environment to monitor certain phenomena such as temperature and humidity. Consequently, objects can be tracked remotely allowing for the communication between the physical and virtual worlds.

The popularity of the IoT notion relies mainly on current technologies: internet, mobile technologies, cloud computing, communication protocols, and embedded sensors to capture the data [3]. In IoT, data is generated by things, so real world objects are considered the core components of the IoT paradigm. Every object has a distinctive identity and can access the network to integrate between both the physical and digital worlds to provide enhanced services to people. IoT can provide device to device, device to people and device to environment information transfer through the integration of information space and physical space [4], [5].

IoT architecture is composed of three levels as shown in Fig. 1.

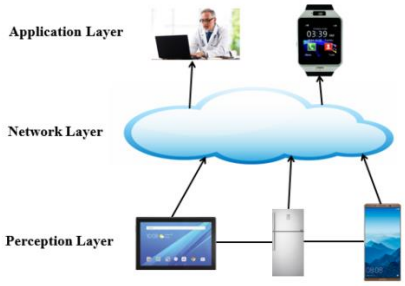

Fig. 1. IoT Architecture (Adapted From [6]).

The topmost layer is the application layer which represents the application service support system. The intermediate layer is the network layer which contains the communication network infrastructure. The bottom layer is the perception layer which comprises the sensor based devices and environmental objects. The captured data from this layer is transferred to the network layer for further processing and analysis [6], [7].

IoT applications generate enormous amounts of data which are characterized by the 5V model

1) *Volume*: huge quantities of generated data.
2) *Variety*: different data types such as structured, semi-structured, and unstructured data.
3) *Velocity*: immense speed of data production and processing.
4) *Veracity*: accuracy and trustiness of the generated data.
5) *Value*: benefits yield from using the data [8].

IoT has numerous applications in different fields such as healthcare, business, smart homes, etc. IoT applications

became extensively used in people's daily lives to make their lives more comfortable [3]. IoT applications are categorized into three main areas:

1) Personal: such as smart homes, telemedicine, and wearables.

2) Social: such as smart grid, smart lighting, and waste management.

3) Business: such as smart farming and smart retail [1], [2].

IoT faces a lot of challenges varying from data management to security threats. Regarding data management, sensors generate enormous amounts of data with various formats so data fusion techniques are required to combine the data. In addition, the IoT environment is vulnerable to a high noise rate since it mainly relies on sensors which possibly be of low power and poor quality [8]. IoT is still in its infancy so it faces a lot of difficulties to have successful employment of different applications. One great challenge is the detection of data anomalies emerging from sensors' data.

II. Motivation

Sensors generate enormous amounts of data that need to be handled efficiently. IoT applications mainly depend on data generated from these sensors, as a result, anomalies can substantially minimize the effectiveness of IoT applications and consequently may lead to inaccurate decisions. Anomaly detection is beneficial because anomalies are doubtful of not being generated by the same methods as the other data points.

The discovery of data anomalies in IoT is a sophisticated task because it is difficult to determine the normal pattern of data as data in the IoT environment is domain dependent [8]. Moreover, multiple sensors continuously generate data to monitor a certain phenomenon so the generated data have various formats.

Actually there are two main causes of data anomalies, namely: an event and noise. An event refers to a specific incident which took place at a certain time interval, whereas noise is just an error, usually because of: poor quality sensors, environmental effects or communication problems. Both event and noise are considered anomalies in terms of having a great deviation from normal data points, but actually they have two different interpretations.

Event detection in IoT is essential since late discovery of certain events such as a fire can cause huge problems. On the other hand, a noise is just considered an error resulting from sensors.

To the best of our knowledge, no work exists answering the question of how to differentiate between an event and noise in IoT. As result, an approach is needed to differentiate between an event and noise since both are considered abnormal points, i.e anomalies so, in this paper, an algorithm is proposed to differentiate between an event and noise. The main contributions are

1) Proposing a novel algorithm for differentiating between an event and noise based on both sensors' values and correlation existence between sensors in the IoT environment.

2) Utilizing the factor of correlation existence between the sensors.

3) Applying the proposed algorithm on a real dataset to evaluate its effectiveness.

The rest of the paper is organized as follows: Section 3 presents the literature work of anomaly detection in IoT. Section 4 presents the categories of anomalies. Section 5 presents the proposed algorithm and experiments. Section 6 presents the conclusion and future work.

III. Related Work

Since the IoT paradigm is still in its beginning, little work investigated the detection of anomalies in this new environment. In [9], the paper presented an approach for detecting data anomalies through utilizing expert knowledge. The proposed approach made use of the possible expected attacks for discovering anomalies through a set of predefined constraints on the data. In [10], a real world simulation prototype was proposed that used IoT smart objects to detect behavioral based anomalies across a simulated smart home. The proposed technique used immunity inspired algorithms to discriminate between normal and abnormal behavioral patterns.

In [11], an unsupervised anomaly detection approach using light switches was presented. The proposed algorithm used a statistical based algorithm using expectation maximization to construct the mixture models. In the proposed technique, an anomaly was correlated with a probability. In [12], a correlation based anomaly algorithm was presented as a predictive maintenance method for compact electric generators. Correlations between sensors were determined by using statistical analysis. Anomalies were detected through analyzing sensors' data and correlation coefficients between sensors.

In [13], a new notion of urban heartbeat which was constructed from sensors' data in the surrounding environment was proposed. Urban Heartbeat collected the contextual information about patterns which occur regularly in the environment. Techniques were developed to find couplings between sensors. Next, quasi periodic patterns were determined from the data. After that, unexpected events which deviate significantly from the normal behavior were discovered.

In [14], air pollution elements were used to discover the unhealthy or anomalous locations in a smart environment. Anomalies were discovered through examining the air quality index, which is a numerical measure used to find out the anomalous locations which goes beyond a specific threshold. Neural networks, neuro fuzzy method, and support vector machines for binary and multi class problems were applied to identify anomalous locations from a pollution database.

IV. Categories of Anomalies

An anomaly/outlier is a data point that greatly differs from the remaining data points, as though it was produced by another approach [15]. There are three main types of anomalies, described as follows:

1) Global/Point anomaly: in a certain dataset, a data point is a global anomaly if it differs substantially from the remaining data points [15]. Global anomalies are considered the easiest type of anomalies to discover and most anomaly detection techniques focus on detecting them.

2) Contextual anomaly: in a particular dataset, a data point is considered a contextual anomaly if it noticeably differs in the defined context [16]. Contextual anomalies are also named as conditional anomalies because they rely on a specific context. As a result, to discover contextual anomalies, the context has to be determined as a core component of the problem definition. In contextual anomaly detection, the attributes of the data points in consideration are categorized into two types:

- Contextual attributes: these features determine the object's context. Context can refer to a time interval or location.
- Behavioral attributes: these attributes refer to the object's characteristics, and are used to determine whether a data point is an anomaly in the context which it exists [15].

3) Collective anomaly: in a certain data set, a subset of data points creates a collective anomaly if the points as a whole vary greatly from the whole dataset. The individual data points may not be anomalies [16].

In this paper, we will focus on detecting global anomalies.

V. Proposed Approach

In this section, the proposed algorithm along with the experiments will be presented. The process of differentiating between an event and noise consists of two main phases:

- The *first* phase detects the anomalies.
- The *second* phase decides whether each anomaly is an event or noise based on the conditions specified in the proposed algorithm.

The process of differentiating between an event and noise is depicted in Fig. 2

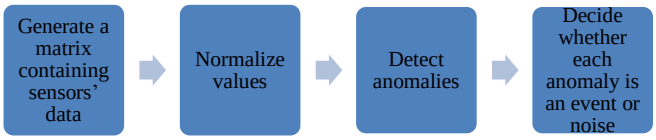

Fig. 2. Process of Detecting Anomaly's Type.

At first, a matrix is generated to include the sensors' data. Then, the data values are normalized. After that, anomalies are detected. At last, the anomaly is either defined as an event or noise. The exact steps of the algorithm will be illustrated in the following subsections.

A. Anomaly Detection

Sensor' data are usually time series data so techniques that fit time series data should be used to detect anomalies. As a result, in this paper, the technique used for anomaly detection is Exponential Moving Average (EMA), also known as an exponentially weighted moving average (EWMA). Exponential moving average is a technique for smoothing time series data using the exponential window function [17].

In the simple moving average, the previous observations are weighted equally, whereas in exponential moving average, exponential functions are used to assign exponentially decreasing weights over time and the weighting for each older data point decreases exponentially [18], that's why EMA is commonly used in analysis of time series data. The advantage of EMA is that it keeps little record of previous data since it focuses on most recent observations, as the most recent data should be given more weight.

In our proposed approach, EMA analyzed whether the value of the attribute being investigated in a given timestamp exceeds a certain threshold. EMA was chosen because it gives more weight to recent observations rather than older ones, so this will help in determining the trend of data and differentiating between an event and noise.

Luminol [19] which is a light weight python library for time series data analysis, was utilized in the experiments. It supports anomaly detection using EMA. The anomaly score was calculated, then the score was compared to a certain threshold to decide whether the data point is an anomaly or not.

Usually, it is recommended to set the threshold based on the statistical principle which states that: to consider a value as an anomaly, it either exceeds $\mu+3\sigma$ or goes below $\mu-3\sigma$ where μ is the mean value and σ is the standard deviation of the attribute under observation [20], so in our experiments, we used this principle to determine the threshold value.

B. Differentiation between an Event and Noise

The following paragraphs will present the algorithm and experiments in details.

1) Data preprocessing:- At first, data need to be preprocessed so min-max normalization was applied on the dataset. Normalization was done through sklearn.preprocessing.MinMaxScaler [21], [22]. Scikit-learn (sklearn) [23] is a free software machine learning library for the Python programming language, and MinMaxScaler is a preprocessing module which scales each value such that it is in the range between zero and one.

2) Proposed algorithm:- Data from sensors can be represented by a data matrix produced by every sensor at each timestamp, denoted as s_{ti}, where s_{ti} refers to the measured value of attribute i at a timestamp t, described as follows in (1)

$$S_{ti} = \begin{bmatrix} S_{11} & S_{12} & \dots & S_{1n} \\ S_{21} & S_{22} & \dots & S_{2n} \\ S_{31} & S_{32} & \dots & S_{3n} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (1)$$

Where t denotes the timestamp, whereas i refers to the sensor number and n denotes the number of sensors.

At any given timestamp, a sensor can be correlated with any other sensor in the surrounding environment. The sensors'

values can be either positively correlated or negatively correlated. To determine whether sensors are correlated or not, a correlation matrix of zeros and ones was constructed to define the correlation between sensors. The correlation value was either zero or one, zero refers to absence of correlation, whereas one refers to presence of correlation (either positively or negatively) between the sensors.

To know if two attributes are correlated or not, we should check the correlation matrix. For example, the average speed and flow of cars in a certain road are negatively correlated because, when the flow (number) of cars increases at a certain timestamp, the average speed of cars decreases. The correlation matrix between these two attributes will be as follows in (2)

$$\begin{array}{c} \\ \text{Flow} \\ \text{Speed} \end{array} \begin{array}{c} \begin{array}{cc} \text{Flow} & \text{Speed} \end{array} \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{array} \quad (2)$$

To use the correlation notion efficiently, only the functionally correlated sensors should be examined since they usually measure the same phenomena. As a result, the values of the functionally correlated sensors can be used in differentiating between an event and noise.

At first, anomalies should be detected using EMA, then the proposed algorithm is applied to differentiate between an event and noise. It detects whether this anomaly is an event or noise. The proposed algorithm is depicted in Fig. 3.

To better illustrate the algorithm, there are four different cases

- s_{ti}, $s_{(t-1)i}$ and $s_{(t+1)i}$ are anomalies.
- s_{ti} is an anomaly whereas $s_{(t-1)i}$ and $s_{(t+1)i}$ are not anomalies.
- s_{ti} and $s_{(t-1)i}$ are anomalies whereas $s_{(t+1)i}$ is not anomaly.
- s_{ti} and $s_{(t+1)i}$ are anomalies whereas $s_{(t-1)i}$ is not anomaly.

```
Given an anomaly value at timestamp t
if s(t-1)i is anomaly then
    if s(t+1)i is anomaly then
        sti ← event
    else
        Check sensors' readings of functionally correlated sensors at timestamp (t)
        if Sensors' readings are anomalies then
            sti ← end of event
        else
            sti ← noise
else if s(t+1)i is anomaly then
    Check sensors' readings of functionally correlated sensors at timestamp (t+1)
    if Sensors' readings are anomalies then
        sti ← event
    else
        sti ← noise
else
    sti ← noise
```

Fig. 3. Proposed Algorithm.

The two main contributions of the proposed algorithm are

1) Utilizing the following timestamp: Most existing anomaly detection algorithms use previous timestamps to discover anomalies, whereas the proposed algorithm used the following timestamp besides the previous timestamp to differentiate between an event and noise. The idea behind using the following timestamp is to wait for more time so that more accurate decisions can be taken since events usually last for a time interval.

2) Using correlation existence between sensors:- The whole dataset was scanned at once to detect anomalies using EMA then, the proposed algorithm was applied to determine whether each anomaly point is an event or noise depending on the specified conditions in the algorithm.

3) Dataset used:- In order to evaluate the performance of the proposed algorithm, it was applied on a real traffic dataset and the accuracy of detecting events and noise was measured. The dataset consisted of 5000 records with 3 attributes and the proportion of anomalies in the dataset was 5%. The dataset presented real time traffic data from the Twin Cities Metro area in Minnesota, collected by the Minnesota Department of Transportation. The Minnesota Department of Transportation captured traffic data on the freeway system throughout the Twin Cities Metro area [24].

The dataset contains occupancy, speed, and flow data for every detector in the Twin Cities Metro area and was collected every 30 seconds. Speed refers to the average value of the speed measurements of individual vehicles over time, whereas flow denotes the number of vehicles passing in a specific point at a certain timestamp. Flow and speed were used, whereas occupancy was not included in the experiments, since occupancy is similar to flow as it represents the percent of time the detection zone of a sensor is occupied by vehicles. These two attributes were selected because they are correlated, i.e., when the flow increases, the speed decreases and vice versa as they exhibit negative correlation.

4) Performance evaluation:- The dataset was used to evaluate the performance of the proposed algorithm. The prediction accuracy of detecting both events and noise was computed. The accuracy was measured as in (3) and (4):

Prediction accuracy of detecting events =

$$\frac{\text{number of correct predictions}}{\text{number of events}} \quad (3)$$

Prediction accuracy of detecting noise =

$$\frac{\text{number of correpredictions}}{\text{number of noise values}} \quad (4)$$

Given that most of the available real data have no class labels, so anomaly labels (both an event and noise) were artificially added to the dataset in order to measure the prediction accuracy of the proposed algorithm.

The prediction accuracy of detecting events and noise is shown in Fig. 4.

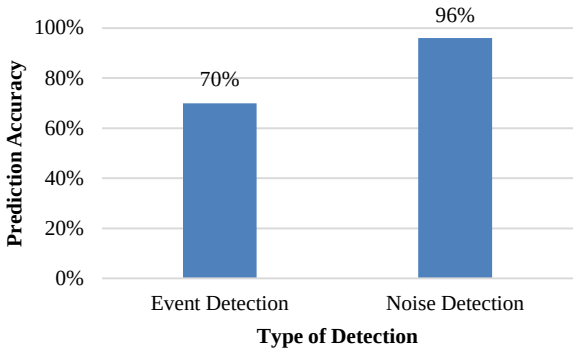

Fig. 4. Prediction Accuracy of Event and Noise Detection.

The proposed algorithm gave promising results especially in detecting noise. The accuracy is higher in detecting noise rather than events, maybe because it is more difficult to detect events, since events' detection involves several factors such as the sensors' values of both the preceding and following timestamps, the values of the functionally correlated attributes, and the nature of event. On the other hand, the noise is just an error resulting from the sensors.

VI. Conclusion and Future Work

IoT is a new paradigm that recently gained popularity. IoT is the integration of physical objects that are attached with software, sensors, and network connectivity, which allow them to capture and transmit data. The IoT paradigm faces numerous challenges ranging from data management to security threats. A substantial challenge is the detection of data anomalies from sensors' data. An anomaly is a data point that greatly varies from the rest of data points. There are two main causes of data anomalies which are: an event and noise. An event denotes an incident which happened at a certain time, whereas noise is just an error. An approach is needed to distinguish between an event and noise since both are considered anomalies so, in this paper, an algorithm was proposed to differentiate between an event and noise in IoT. Also, the effectiveness of the algorithm was tested through experiments.

In future work, we will explore how to enhance the accuracy of the algorithm in detecting events. Also, the algorithm will be applied on other datasets in different domains.

References

[1] S. Ray, Y. Jin, and A. Raychowdhury, "The changing computing paradigm with internet of things: a tutorial introduction," IEEE Design and Test, vol. 33, no. 2, pp. 76–96, April 2016.

[2] S. Elbouanani, M. A. E. Kiram, and O. Achbarou, "Introduction to the internet of things security: standardization and research challenges," in *Proc. IAS*, Marrakech, 2015, pp. 32–37.

[3] S. Kraijak and P. Tuwanut, "A survey on internet of things architecture, protocols, possible applications, security, privacy, real-world implementation and future trends," in *Proc. ICCT*, Hangzhou, 2015, pp. 26–31.

[4] Z. Yue, W. Sun, P. Li, M. U. Rehman, and X. Yang, "Internet of things: architecture, technology and key problems in implementation," in *Proc. CISP*, Shenyang, 2015, pp. 1298–1302.

[5] S. Nalbandian, "A survey on internet of things: applications and challenges," in *Proc. ICTCK*, Mashhad, 2015, pp. 165–169.

[6] D. Rose, Enchanted Objects: Design, Human Desire, and the Internet of Things, 1st ed., New York: Simon & Schuster, 2014.

[7] W. Z. Khan, H. M. Zangoti, M. Y. Aalsalem, M. Zahid, and Q. Arshad, "Mobile RFID in internet of things: security attacks, privacy risks, and countermeasures," in *Proc. ICRAMET*, Jakarta, 2016, pp. 36–41.

[8] I. Butun, B. Kantarci, and M. Erol-Kantarci, "Anomaly detection and privacy preservation in cloud-centric internet of things," in *Proc. ICCW*, London, 2015, pp. 2610–2615.

[9] V. A. Desnitsky, I. V. Kotenko, and S. B. Nogin, "Detection of anomalies in data for monitoring of security components in the internet of things," in *Proc. SCM*, St. Petersburg, 2015, pp. 189–192.

[10] B. Arrington, L. Barnett, R. Rufus, and A. Esterline, "Behavioral modeling intrusion detection system (BMIDS) using internet of things (IoT) behavior-based anomaly detection via immunity-inspired algorithms," *in Proc. ICCCN*, Hawaii, 2016, pp. 1–6.

[11] C.-W. Ho, C.-T. Chou, Y.-C. Chien, and C.-F. Lee, "Unsupervised anomaly detection using light switches for smart nursing homes," in *Proc. DASC*, Auckland, 2016, pp. 803–810.

[12] P. Zhao, M. Kurihara, J. Tanaka, T. Noda, S. Chikuma, and T. Suzuki, "Advanced correlation-based anomaly detection method for predictive maintenance," in *Proc. ICPHM*, Seattle, 2017, pp. 78–83.

[13] S. A. Hasnain and R. Jafari, "Urban heartbeat: From modelling to applications," in *Proc. SMARTCOMP*, Hong Kong, 2017, pp. 1–8.

[14] R. Jain and H. Shah, "An anomaly detection in smart cities modeled as wireless sensor network," in *Proc. IConSIP*, Nanded, 2016, pp. 1–5.

[15] J. Han, M. Kamber, and J. Pei, "Outlier detection," in Data Mining: Concepts and Techniques, 3rd ed., Netherlands: Elsevier, 2012, pp. 544–548.

[16] D. Hand, H. Mannila, and P. Smyth, "Measurement and data," in Principles of Data Mining, Cambridge: The MIT Press, 1st ed., 2001, pp. 35–36.

[17] D. R. Anderson, D. J. Sweeney, and T. A. Williams, "Descriptive statistics: numerical measures," in Essentials of Modern Business Statistics, 3rd ed., Boston: Cengage Learning, 2012, pp.147–148.

[18] "Exponential smoothing," *Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Exponential_smoothing. [Accessed: 1-Novmeber-2018].

[19] Linkedin, "linkedin/luminol," *GitHub*. [Online]. Available: https://github.com/linkedin/luminol. [Accessed: 1-November-2018].

[20] H.-P. Kriegel, P. Kröger, and A. Zimek, "Outlier detection techniques," in *Proc. SDM*, Columbus, 2010, pp. 1–73.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: machine learning in python," *JMLR*, vol. 12, pp. 2825–2830, October 2011.

[22] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. Vanderplas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *Proc. ECML PKDD*, Prague, 2013, pp. 108–122.

[23] "Scikit-learn," *Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Scikit-learn. [Accessed: 10-November-2018].

[24] "Mn/DOT Traveler Information," *Minnesota Department of Transportation*. [Online]. Available: http://www.dot.state.mn.us/tmc/trafficinfo/developers.html. [Accessed: 11-November-2018].