

Article

# Multivariate-Time-Series-Driven Real-time Anomaly Detection Based on Bayesian Network

Nan Ding \* , Huanbo Gao, Hongyu Bu, Haoxuan Ma and Huaiwei Si

School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China; huanboliuyue@mail.dlut.edu.cn (H.G.); bhy650@mail.dlut.edu.cn (H.B.); haoxuan19950916@mail.dlut.edu.cn (H.M.); sihuaiwei@mail.dlut.edu.cn (H.S.)

\* Correspondence: dingnan@dlut.edu.cn

Received: 25 August 2018; Accepted: 4 October 2018; Published: 9 October 2018



**Abstract:** Anomaly detection is an important research direction, which takes the real-time information system from different sensors and conditional information sources into consideration. Based on this, we can detect possible anomalies expected of the devices and components. One of the challenges is anomaly detection in multivariate-sensing time-series in this paper. Based on this situation, we propose RADM, a real-time anomaly detection algorithm based on Hierarchical Temporal Memory (HTM) and Bayesian Network (BN). First of all, we use HTM model to evaluate the real-time anomalies of each univariate-sensing time-series. Secondly, a model of anomalous state detection in multivariate-sensing time-series based on Naive Bayesian is designed to analyze the validity of the above time-series. Lastly, considering the real-time monitoring cases of the system states of terminal nodes in Cloud Platform, the effectiveness of the methodology is demonstrated using a simulated example. Extensive simulation results show that using RADM in multivariate-sensing time-series is able to detect more abnormal, and thus can remarkably improve the performance of real-time anomaly detection.

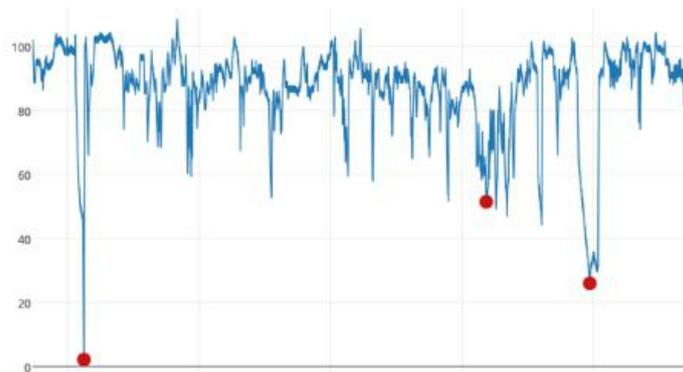
**Keywords:** multivariate-sensing time-series; anomaly detection; hierarchical temporal memory; bayesian network

## 1. Introduction

The system condition monitoring associated with virus invasion, failed sensors, and improperly implemented controls plagues many automated information system, such as wireless sensor networking, vehicular networking, and industrial system. The real-time anomaly detection for system condition monitoring has significant and practical applications, which uses the information coming in real-time from different sensors and other condition information sources and tries to detect possible anomalies in the normal condition and behaviour expected of its devices or components. Anomalies can be spatial, known as spatial anomalies, which means the values are outside the typical range like the 1st and 3rd anomalies in Figure 1, which is time series data collected from temperature sensors in a factory (The anomaly in Figure 1 is marked in red). Also anomalies can be temporal, known as temporal anomalies. The values are not out of typical range but the sequence generated from which is anomalous, like the second anomaly in Figure 1 [1].

At present, there is extensive work on anomaly detection techniques looking for individual objects that are different from normal objects. Some of them mainly focus on the anomaly detection in univariate time series (UTS). Holt-Winters is one of the anomaly detection methods which can detect spatial anomalies and it is widely implemented for commercial applications [2]. The Skyline project provides an open-source implementation of a number of statistical techniques for anomaly detection in streaming data [3]. Autoregressive Integrated Moving Average Model (ARIMA) is a

general technique for modeling temporal data with seasonality which can perform temporal anomaly detection in complex scenarios [4]. Bayesian change point methods are a nature method which segments time series and can be used for online anomaly detection [5,6]. Twitter released its own open-source anomaly detection algorithms for time series data, it is capable of detecting spatial and temporal anomalies, and has gotten a relatively high score in the Numenta Anomaly Benchmark (NAB) scoring mechanism [7,8]. In addition, there are a number of model-based methods applied to specific fields, examples include detection for cloud data center temperatures [9], ATM fraud detection [10], anomaly detection in aircraft engine measurements [11], and some excellent work based on an accurate forecasting solutions with application to the water sector [12,13] so on.



**Figure 1.** Spatial anomalies and temporal anomalies.

However, in a complex system, compared to the anomaly detection in UTS, it brings richer system information using multivariate time series (MTS), which are captured from the sensors and condition information sources. For the methods of anomaly detection in MTS, there are two main categories, one is to use the method of detecting anomaly after dimension reduction. Existing dimensionality reduction methods, such as PCA (Principal component analysis) dimensionality reduction method used for MTS and the linear dimensionality reduction method used for MTS based on common principle component analysis, are processing the principle time series according to the anomaly detection method for UTS [14]. Another method is to take the sliding window as a tool for dividing MTS, and then detect the subsequences. For example, one method is to calculate each covariance matrix of subsequence based on Riemannian manifolds and make the covariance matrix as the descriptor, Riemannian distance as the similarity measure to calculate the distance between the covariance matrix. And it can show the existence of abnormal intuitively through the distribution of the covariance matrix and its visualization [15,16]. These methods could meet the requirements for anomaly detection in MTS, but due to the lack of consideration of the inherent relevance in MTS, actual effect and accuracy are required to be improved. In order to improve the efficiency of the algorithm, this paper introduces the concept of health factor  $\alpha$  which means whether the system is running well enough or not. Introducing health factor can greatly reduce the cost of RADM in a healthy system.

In this paper, we propose a real-time anomaly detection algorithm in MTS based on Hierarchical Temporal Memory (HTM) and BN (Bayesian Network). The remainder of the paper is structured as follows. Section 2 provides a brief introduction to the performance problems and scenario of anomaly detection in MTS, while Section 3 is dedicated to presenting the methodology of HTM. The proposed RADM is discussed in Section 4. Then, in Section 5, to demonstrate the principles and efficacy, results are presented comparing RADM with HTM. Finally, in Section 6, concluding remarks and possible extensions of the work are discussed.

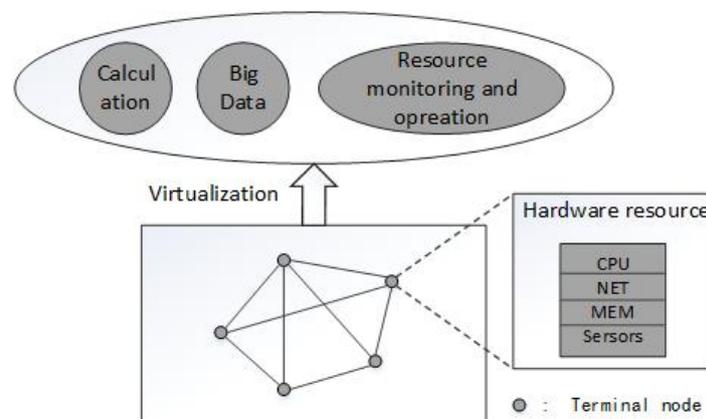
The contributions of this paper can be summarized as follows:

1. This paper proposes an anomaly detection framework based on multivariate-sensing time-series data to achieve real-time anomaly detection, and improve the performance of anomaly detection.

2. This paper introduces the concept of health factor  $\alpha$  to describe the health of the system, and further more, greatly improve the detection efficiency of health systems.
3. RADM combines HTM with naive Bayesian network to detect anomalies in multivariate-sensing time-series, and get better result compared with the algorithm just work in univariate-sensing time-series.

## 2. Performance Problems and Scenario

It has been proved that cloud computing could help by offering on-demand and scalable storage, as well as processing services that can scale to the requirements of IoT and automated industry system. In these applications, Cloud Platform virtualizes physical devices (hosts, switches, storage and sensors, etc.) based on virtualization, and by virtualizing the infrastructure engine integrates the virtual computing, storages and networks resources into a centralized resource pool. And this resource pool is delivered to the upper application system in an automated and self-service way [17,18]. The structure of Cloud Platform is shown in Figure 2.



**Figure 2.** The structure of Cloud Platform.

At present, related works utilize system parameters to characterize the real-time state of the system, and implement the real-time detection for its state. In order to perform the anomaly detection in MTS, based on the real-time monitoring cases of the system states of terminal nodes in Cloud Platform, we virtualize the node system state into CPU, NET and MEM parameters, as shown in Figure 2, and use them as anomaly detection sequences to characterize the system state. We describe the system state as follow:

$$S(t) = (X(t), Y(t), Z(t)) \quad (1)$$

where,  $X(t)$  represents the time series data of CPU,  $Y(t)$  is the time series data of NET, and  $Z(t)$  means the time series data of MEM. Since NET accounts for a large proportion in the reasons of system anomalies, we set NET as the principle time series, and it is utilized in anomaly detection in UTS which is used as a comparison in the experiment.

## 3. UTS and HTM

### 3.1. HTM Cortical Learning Algorithm

HTM is a machine learning method designed to capture the structure and operating characteristics of the new cerebral cortex. HTM is essentially a memory-based system, and the HTM network relies on a large number of pattern sequences stored in it, and it is trained by a large number of time series data. The HTM network consists of hierarchical regions, and each of the regions represents a level

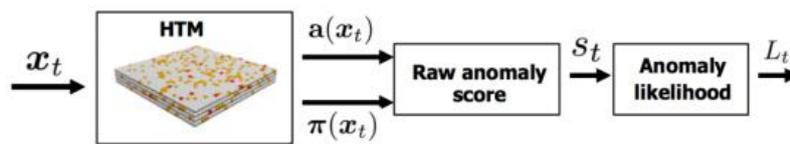
in the hierarchy. Hierarchy information will continue to converge with the rise of the level, and also diverge with the decline of the level. The spatial pooler and the temporal pooler are used in HTM to learn and predict from the input data. The HTM region contains a columnar region consisting of cells, and through the spatial pooler each cell produces a sparse distributed representation, which is used to record the active state of cells. Then the temporal pooler can discover and learn pattern from the list of active columnar regions calculated from the spatial pooler, and serialize it for prediction [19].

### 3.2. Anomaly Detection in UTS Based on HTM

HTM formalizes the process as follows. It lets the vector  $x_t$  represent the state of a real-time system at time  $t$ . The HTM model receives a continuous stream of inputs:

$$\dots, x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}, \dots \quad (2)$$

In practical applications, the statistics of the system can change dynamically, real-time learning and training are needed to perform the detection of new anomalies. HTM is a learning algorithm which appears to match the above constraints and has been shown to work well for prediction tasks [20], however, it does not directly output the anomaly score. In order to perform anomaly detection, two different internal representations are used in the HTM. As shown in Figure 3 [1]. Given an input  $x_t$ , the vector  $a(x_t)$  is a sparse binary code representing the sparse distributed representation of the current input. Also an internal state vector  $\pi(x_t)$  is used to represent a prediction for  $a(x_{t+1})$ , that is, a prediction of the next input  $x_{t+1}$ . The prediction vector incorporates inferred information about current sequences, which is dependent on the current detected sequence and the current inferred position of the input in the sequence, and this means different inputs will lead to different predictions.



**Figure 3.** The primary functional steps in HTM algorithm.

However,  $a(x_t)$  and  $\pi(x_t)$  do not directly represent anomalies. In order to create a robust anomaly detection system, HTM introduces two additional steps. Firstly a raw anomaly score is computed from the two sparse vectors. Then HTM computes an anomaly likelihood value which is thresholded to determine whether the system is anomalous.

#### 3.2.1. Computing the Raw Anomaly Score

In order to measure the deviation between the predicted input and the actual input of the model, a raw anomaly score is computed from the intersection between the predicted and actual sparse vectors. At time  $t$  the raw anomaly score  $s_t$  is given as:

$$s_t = 1 - \frac{\pi(x_{t-1}) \cdot a(x_t)}{|a(x_t)|} \quad (3)$$

The raw anomaly score will be 0 if the current input is perfectly predicted, 1 if it is completely unpredicted, or somewhere between 0 and 1 according to the similarity between the input and the prediction.

Because of the continuous learning nature of HTM, the changes of the underlying system can be handled gracefully. If there is a shift in the behavior of the system, the anomaly score will be high at the point of this shift, however, as the HTM model adapts to the “new normal”, the anomaly score will degrade to zero automatically [1].

### 3.2.2. Computing the Anomaly Likelihood

HTM uses the distribution of the anomaly scores to calculate the anomaly likelihood which defines how anomalous the current state is based on the prediction history of the HTM model. HTM maintains a window of the last  $W$  raw anomaly scores and models the distribution of the anomaly scores as a rolling normal distribution, where the sample mean  $\mu_t$  and variance  $\sigma_t$  are continuously updated from the previous anomaly scores, Using  $k$  as the number of windows, as follows:

$$\mu_t = \frac{\sum_{i=0}^{W-1} s_{t-i}}{k} \quad (4)$$

$$\sigma_t^2 = \frac{\sum_{i=0}^{W-1} (s_{t-i} - \mu_t)^2}{k-1} \quad (5)$$

Then HTM evaluates a recent short-term average of the anomaly scores, and thresholds the Gaussian tail probability (Q-function [21]) to decide whether or not to declare an anomaly. The anomaly likelihood is defined as the complement of the tail probability:

$$L_t = 1 - Q\left(\frac{\tilde{\mu}_t - \mu_t}{\sigma_t}\right) \quad (6)$$

where:

$$\tilde{\mu}_t = \frac{\sum_{i=0}^{W'-1} s_{t-i}}{k} \quad (7)$$

$W'$  is a window for a short-term moving average, where  $W' \ll W$ . HTM sets the threshold  $L_t$ , if it is very close to 1, an anomaly will be reported:

$$anomaly \equiv L_t \geq 1 - \epsilon \quad (8)$$

HTM sets  $Q$  as an array of the tail probability for the standard normal distribution, which preserves the probability corresponding to the mean. In order to facilitate the calculation, HTM divides the range of values of  $\mu$  in the normal distribution  $[\sigma, 3.5\sigma]$  into 70 intervals equidistantly, so that it sets the tail probability to 71 values and puts them into an array  $Q$ , as shown in Table 1.

**Table 1.** List of the tail probability values.

Tail Probability Elements	Values
Q[0]	0.500000000
Q[1]	0.460172163
Q[2]	0.420740291
...	...
Q[68]	0.000000000005340
Q[69]	0.000000000002653
Q[70]	0.000000000001305

In the process of thresholding  $L_t$ , thresholding the tail probability makes the number of alerts have an inherent upper limit. In addition, since  $\epsilon$  is very close to 0, it would be unlikely to get alerts with probability much higher than  $\epsilon$ , which also imposes an upper limit on the number of false positives.

## 4. MTS and RADM

In order to meet the requirements for anomaly detection in MTS and consider the inherent relevance between MTS, we utilize a BN model to analyze the validity of MTS, and propose a corresponding anomaly detection algorithm in MTS, RADM.

#### 4.1. Bayesian Network Analysis

##### 4.1.1. Bayesian Network

BN is a combination of Graph Theory and Probability Theory, and consists of a directed acyclic graph and the probability of each node. This directed acyclic graph is called the BN structure, the probability of each node is called the BN parameter, BN gets the BN parameters through structural learning or parameter learning. The basic formula of the graphical network is the Bayesian formula [22]. According to the Bayesian formula, a sample  $x$  is known, and the probability that this sample belongs to the category  $y$  is:

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)} \quad (9)$$

where  $x$  is the sample characteristic variable, that is, the known quantity. According to the Total Probability Theorem, expansion is shown as follows:

$$P(y = c_k|x) = \frac{P(x_1, x_2, \dots, x_i|y = c_k)P(y = c_k)}{\sum_k P(y = c_k)P(x_1, x_2, \dots, x_i|y = c_k)} \quad (10)$$

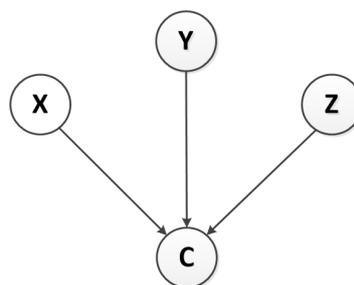
where  $c_k$  represents the  $k$ -th class of  $y$  and  $i$  is the number of characteristic variables of  $x$ ,  $x_i$  is the value corresponding to the  $i$ -th characteristic variable. Let  $M$  be the characteristic variable dimension of  $x$  (i.e., the total number of characteristic variables). If characteristic variables in the sample  $x$  are independent of each other, the above formula can be simplified as:

$$P(y = c_k|x) = \frac{\prod_{i=1}^M P(x_i|y = c_k) * P(y = c_k)}{\sum_k P(y = c_k) \prod_{i=1}^M P(x_i|y = c_k)} \quad (11)$$

##### 4.1.2. Naive Bayesian Network in MTS

We choose to build a Naive Bayesian classifier with multivariate parameters. Naive Bayesian is insensitive to missing data, and it has minimal error rate compared to other classification methods [23].

The topological structure of the Naive Bayesian model is shown in Figure 4,  $C$  represents the classification mark, that is, the output sequence,  $C = \{\text{normal, anomaly}\}$ . We mark the state variables of  $X(t), Y(t), Z(t)$  as  $X, Y$ , and  $Z$  respectively, that is, the discrete values corresponding to the anomaly likelihood of the ternary time series. Because the nodes of the Naive BN are independent of each other, its classifier satisfies the Equation (11).



**Figure 4.** The Topology of Bayesian Network.

For the same group of state variables  $[X, Y, Z]$ , the denominator of the formula is the same, so only the numerator is needed to be compared and we get:

$$c(x) = \operatorname{argmax} P(X, Y, Z|c)P(c) \quad (12)$$

When using a set of  $[X, Y, Z]$  data as a training sample (training set), the following Bayesian formula is used to calculate the posterior distribution of the target variable:

$$c(x) = \operatorname{argmax} \prod_{i=1}^3 P(a_k|c)P(c) \quad (13)$$

where  $a_k$  represents three attribute variables  $[X, Y, Z]$ , so  $k$  can be the value of 1,2,3.

The following formula is used to calculate  $P(c)$ :

$$P(c) = \frac{\sum_{i=1}^n I(c = c_j)}{n} \quad (14)$$

where,  $n$  is the number of training samples,  $c_j$  marks the  $j$ -th category of  $c$ ,  $I(c = c_j)$  is the instruction function, when the equation in brackets is true, its value is 1, otherwise 0. Then it is assumed that the  $k$ -th characteristic in the three characteristics has  $l$  values, and one of the values is assumed to be  $a_{jl}$ , the following formula is used to calculate  $P(a_k|c)$ :

$$P(a_k|c) = \frac{\sum_{i=1}^n I(x_i^j = a_{jl}, c = c_j)}{\sum_{i=1}^n I(c = c_j)} \quad (15)$$

After the above calculation, we can get the probability that a set of eigenvalues belongs to a class and then perform the classification.

#### 4.2. RADM

Based on the above HTM algorithm and Naive Bayesian model, we design a real-time anomaly detection algorithm in MTS, RADM. The anomaly detection flowchart of RADM is described in Figure 5.

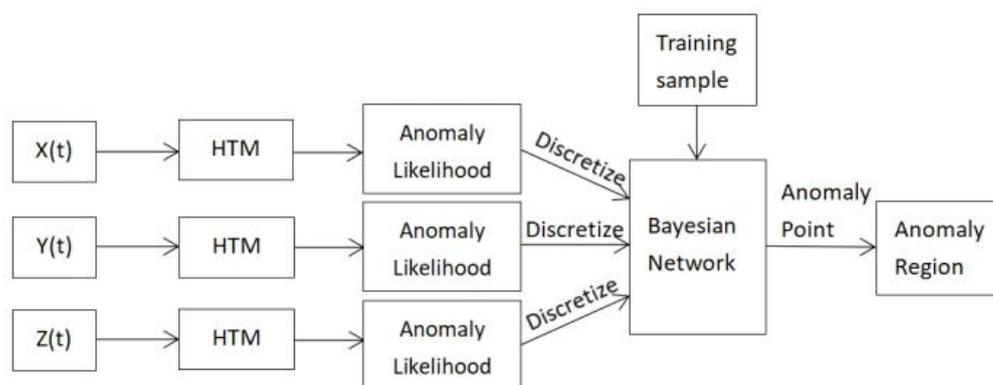


Figure 5. RADM flowchart.

##### (1) HTM Algorithm

The ternary sequences  $X(t)$ ,  $Y(t)$  and  $Z(t)$  need to be detected separately, three HTM networks are used to study and predict three sequences, we get the anomaly likelihood through HTM algorithm, and a list of anomaly likelihood is composed of the anomaly likelihood in three sequences. The specific process can be found in Section 3.

##### (2) Discretization

Because in the calculation of anomaly likelihood the values of the anomaly likelihood are also divided into 70 intervals, in order to facilitate the study and reduce the amount of data and calculation

of training sample in parameter learning, we need to discretize the anomaly likelihood. By analyzing the likelihood data we know that the anomaly likelihood of each variable is related to the standard normal distribution and the range of the low anomaly likelihood is large but the effect to the anomaly detection is negligible. After the experimental tests, combined with the short-term sliding average we use the equivalent interval method to get the threshold intervals and the discrete values, as shown in Table 2.

**Table 2.** The discrete values of MTS.

Time Series	Threshold Intervals	Discrete Values
Input	0-0.890401416	1
	0.890401416-0.997300202	2
	0.997300202-0.99998220767	3
	0.99998220767-1	4
Output	Normal	1
	Anormal	2

### (3) Bayesian Network

We use the method of parameter learning to build Naive BN. After we know the Naive Bayesian structure, the corresponding weights are assigned to each time series according to the prior knowledge. Since NET is the principle time series in this experiment, we need to increase the weight of NET when training BN, and then builds the training sample based on these weights to train BN. We use Matlab to implement BN and after the discretization the junction tree method is used to infer the anomaly classification, so that we get the list of anomaly regions.

The relevant pseudo-code of RADM is shown as follow Algorithm 1:

---

#### Algorithm 1: RADM.

---

```

1: Input:  $S(t)$ ;
2: Output:  $ARmuti(t)$ ; // Anomaly regions in MTS
3: while 1 do
4:    $ALx(t) = HTM(S(t).X(t))$ ;
   // The list of anomaly likelihood in X (t) through HTM
5:    $ALy(t) = HTM(S(t).Y(t))$ ;
6:    $ALz(t) = HTM(S(t).Z(t))$ ;
7:    $D(X, Y, Z) = Discrete(ALx(t), ALy(t), ALz(t))$ ;
   // Discretization
8:    $ARmuti(t) = Bayesian(D)$ ; // BN inference
9: end while

```

---

Since each exception likelihood obtained by HTM corresponds to a point in a time series, in the actual test, there will be continuous anomaly points. It is obviously unreasonable to judge the number of anomalies based on anomaly points. Therefore we use an anomaly region to represent an anomaly. The anomaly region is divided according to the continuous situation between the anomaly points, the formal definition is as follows:

If the distance between an anomaly point A1 and another anomaly point A2 is less than the division distance S, then we put A1 and A2 into the same anomaly region, on the contrary, they belong to different anomaly regions. The division distance S is also called the division window.

#### (4) Health Factor

So as to improve the efficiency of the algorithm, this paper introduces the concept of health factor  $\alpha$ , which is defined as Formulas (16) and (17).  $S_n$  is the anomaly score calculated by HTM. Health factor means whether the system is running well enough or not. The reason for introducing health factor is that the ratio of anomalies in different systems is different. For a system with extremely low probability of anomalies in daily operation, if we import all data into the Naive Bayesian model for joint detection, it will waste a lot of system resources and also affect the temporal performance of the algorithm. In this paper, only when  $\alpha$  is higher than the threshold  $\theta$ , the data is imported into the Naive Bayesian model for joint detection, which can effectively improve the temporal performance of the algorithm. The threshold  $\theta$  can be dynamically selected between 0 and 1 depending on the system state. When  $\theta$  is equal to 0 ( $\theta = 0$ ), it means that all the data is imported into the Naive Bayesian model for joint determination, which corresponds to the condition where the system state is extremely unhealthy. When  $\theta$  is equal to 1 ( $\theta = 1$ ), it represents the system is completely normal, and no matter what value of  $\alpha$  is, it is not necessary to enter the Naive Bayesian model for joint determination.

$$H = (S_1, S_2, S_3, \dots, S_n) \quad (16)$$

$$\alpha = \|H\| = \sqrt{(S_1 - \bar{S}_1)^2 + (S_2 - \bar{S}_2)^2 + \dots + (S_n - \bar{S}_n)^2} \quad (17)$$

## 5. Experimental Simulation and Performance Analysis

Through experiments, we verify the effect of anomaly detection in MTS based on RADM, simultaneously, adding the experiment of anomaly detection in UTS based on HTM algorithm as a comparison. By comparing the results of these two experiments, we analyze the advantages of RADM for anomaly detection, and how the anomaly detection changes from UTS to MTS.

### 5.1. Simulation Environment and Parameter Settings

As for the implementation of HTM algorithm, we select the htm.java code library published by Numenta on Github in 2017 [24], runtime environment is Java (TM) SE Runtime Environment 1.8.0. We use FullBNT-1.0.4 as Bayesian tool, compiler environment is Matlab 2016b.

In the anomaly detection in UTS, window  $W'$  for sliding short-term average should be much smaller than the sliding time window  $W$ , in our experiment,  $W$  is set to 100 and  $W'$  is set to 5, anomaly threshold is set to 0.99998220767. In addition, we use an anomaly region to represent an anomaly, the size of the division window of the anomaly region is set to 10. In order to avoid the influence of different health factor  $\alpha$  on the performance of the algorithm, experiment makes  $\alpha = 0$ , that is, the overall data in the dataset needs to enter the Naive Bayesian model for analysis.

### 5.2. Data Sample Description

Numenta sets the Numenta Anomaly Benchmark (NAB) which is an open-source framework, to compare and evaluate algorithms used to detect anomalies in real-time streaming data. Its NAB dataset contains real world tag data from multiple domains, which includes the normal time-series data and the labeled anomalous time-series data, and it has been proved to be effective in the assessment of the anomaly detection in UTS [7].

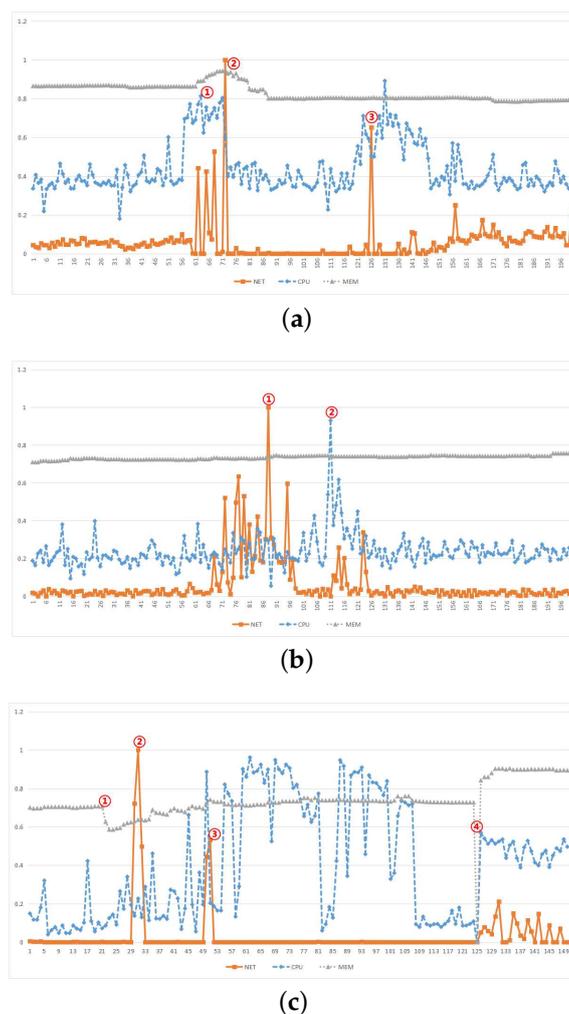
In order to perform the real-time anomaly detection in MTS, we refer to the NAB dataset and its requirements, by means of stress-testing approaches we let  $X(t)$ ,  $Y(t)$  and  $Z(t)$  contain anomalous subsequences, and with the help of the open-source tool sigar1.6.4 java version we program and collect  $X(t)$ ,  $Y(t)$  and  $Z(t)$  in the computer. We use them as data samples input into RADM algorithm.

### 5.3. Performance Analysis and Comparison

#### 5.3.1. Relevance Analysis

RADM can detect the anomalies in the time series effectively through the multi-parameter auxiliary judgment. For a complex system, the variables are often interrelated, this relationship can be reflected in the MTS. And BN has the advantage of dealing with uncertainties, it can show this relationship intuitively and simply in the form of probability.

As shown in Figure 6, from the three points in Figure 6a the relevance of the data is reflected, we can see that in the vicinity of these points, the NET changes will cause changes of CPU and MEM, which means when an anomaly occurs in the associated data, other time series also have a relatively large probability of anomaly. The two points in Figure 6b reflects the necessity of anomaly detection in MTS. Shortly after the peak of NET occurs, the CPU reaches peak, making the range of anomaly regions increase, which is not possible for anomaly detection in UTS. From the Figure 6c we can see when only the NET time series is detected, the range of the anomaly region is between the 2nd and 3rd points, but because of the detection of other anomaly time series, the range of anomaly region is extended to the scope of the 1st and 4th points, which also enables that the anomaly can be detected in advance. The expansion of the anomaly region helps to understand the anomaly information and make more comprehensive decisions.



**Figure 6.** Analysis of the relevance between different dimensions: (a) Establish relevance based on MEM, (b) Independent CPU and NET, (c) Mixed situation.

Definitely, the anomaly detection in UTS is the basis of RADM. However, as the dimension increases, the MTS can play a greater role in anomaly detection in a complex system and reflect the relevance between data, and this relevance is the embodiment of data validity.

### 5.3.2. Accuracy Analysis

We perform anomaly detection in UTS based on HTM algorithm and anomaly detection in MTS based on RADM separately. The two experiments use the same data, which is, when performing anomaly detection in UTS, only detecting  $Y(t)$  in the MTS. And ternary time series need to be detected in the anomaly detection of RADM.

The experimental results are shown in Table 3. It can be seen from the 1st, 4th group data that RADM can detect more anomaly regions than the anomaly detection in UTS. However, the 5th, 6th group data show that the anomaly regions detected by RADM is less than the anomaly detection in UTS, which does not mean the capability of RADM is low, in contrast, the number of anomalies detected by RADM is always greater than the anomalies detected by the anomaly detection in UTS, that is because when the anomaly points are divided into anomaly regions, RADM can detect more anomaly points, which makes some of the anomaly regions that should be separated in the case of anomaly detection in UTS are merged together in the case of anomaly detection in MTS, so that the number of anomaly regions is reduced. In general, the average precision of RADM is 0.706. By contrast, the average precision of HTM is 0.642.

**Table 3.** Performance comparison.

Group	Sequence Length	Algorithm	Number of Anomaly Regions	Precision
1	4806	HTM	18	0.613
		RADM	27	0.675
2	4688	HTM	22	0.667
		RADM	22	0.724
3	4317	HTM	27	0.603
		RADM	27	0.632
4	4216	HTM	21	0.659
		RADM	22	0.757
5	4813	HTM	32	0.724
		RADM	26	0.775
6	4127	HTM	17	0.655
		RADM	15	0.710
7	4342	HTM	14	0.584
		RADM	14	0.656
8	4334	HTM	20	0.653
		RADM	22	0.749
9	4656	HTM	21	0.622
		RADM	25	0.695
10	4357	HTM	28	0.707
		RADM	27	0.778
11	4442	HTM	16	0.575
		RADM	20	0.644
12	4553	HTM	22	0.643
		RADM	22	0.686

Compared with the running time of the two experiments, we can see that the running time of RADM is about three times of the anomaly detection algorithm based on HTM, even so, RADM also

has a high computational efficiency. That is because for every 4000 time series it only costs about 65 s, which has been able to meet the requirements of real-time anomaly detection.

## 6. Conclusions

Anomaly detection is an important research direction. Even though one of the terminal nodes performs anomalously, it will gradually expand the impact and threaten the entire system. RADM algorithm proposed in this paper, which combines HTM algorithm and BN effectively, can be applied to the anomaly detection in MTS in complex system. Compared to UTS, the analysis for MTS can detect anomalies in the system effectively. First of all, we use HTM algorithm to detect anomalies in UTS and get fine detection results as well as excellent time performance. Secondly, we combine HTM algorithm and BN together to perform anomaly detection in MTS effectively without dimension reduction. Some anomalies that be left out in UTS can be detected by this means and detection accuracy is improved. Lastly, in order to improve the efficiency of the algorithm, we introduce the concept of health factor  $\alpha$  to describe whether the system is healthy or not. This method can greatly improve the performance of the algorithm in the health system. Extensive simulation results show that using RADM algorithm to perform anomaly detection in MTS can achieve better result than just in UTS. Furthermore, it will remarkably improve the performance of real-time anomaly detection in many domains. Our future work will include optimizing our algorithm further and improving the detection accuracy. We also attempt to build stronger correlations between multiple variables using other models and apply our algorithms to other areas.

**Author Contributions:** Conceptualization, N.D.; Methodology, N.D.; Resources, N.D.; Analysis, N.D.; Writing-Original Draft, H.B.; Writing-Review and Editing, N.D.; Supervision, N.D.; Funding Acquisition, N.D.; Software, H.B. and H.G.; Validation, H.M.; Visualization, H.S.; Project Administration, N.D.; Preparation, N.D. and H.G. and H.B.

**Funding:** This work was supported in part by the National Science Foundation of China No. 61471084 and the Open Program of State Key Laboratory of Software Architecture No. SKLSA2016B-02.

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

## References

1. Ahmad, S.; Purdy, S. Real-Time Anomaly Detection for Streaming Analytics. *arXiv* **2016**, arXiv:1607.02480 .
2. Szmít, M.; Szmít, A. Usage of modified holtwinters method in the anomaly detection of network traffic: Case studies. *J. Comput. Netw. Commun.* **2012**, *2012*, 192913.
3. A Stanway. Etsy Skyline. Available online: <https://github.com/etsy/skyline> (accessed on 9 October 2018).
4. Bianco, A.M.; Garcia Ben, M.; Martinez, E.J.; Yohai, V.J. Outlier detection in regression models with ARIMA errors using robust estimates. *J. Forecast.* **2010**, *20*, 565–579.
5. Adams, R.P.; MacKay, D.J. Bayesian Online Changepoint Detection. *arXiv* **2007**, arXiv:0710.3742v1.
6. Tartakovsky, A.G.; Polunchenko, A.S.; Sokolov, G. Efficient Computer Network Anomaly Detection by Changepoint Detection Methods. *IEEE J. Sel. Top. Sign. Process.* **2013**, *7*, 4–11. [[CrossRef](#)]
7. Lavin, A.; Ahmad, S. Evaluating Real-time Anomaly Detection Algorithms—the Numenta Anomaly Benchmark. In Proceedings of the IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 9–11 December 2015; pp. 38–44.
8. Kejariwal, A. Twitter Engineering: Introducing practical and robust anomaly detection in a time series. Available online: <http://bit.ly/1xBbX0Z> (accessed on 6 January 2015).
9. Lee, E.K.; Viswanathan, H.; Pompili, D. Model-based thermal anomaly detection in cloud datacenters. In Proceedings of the International Conference on Distributed Computing in Sensor Systems Model-based thermal anomaly detection in cloud datacenters, Cambridge, MA, USA, 20–23 May 2013.
10. Klerx, T.; Anderka, M.; Büning, H.K.; Priesterjahn, S. Model-Based Anomaly Detection for Discrete Event Systems. In Proceedings of the IEEE 26th International Conference on Tools with Artificial Intelligence, Limassol, Cyprus, 10–12 November 2014.

11. Simon, D.L.; Rinehart, A.W. A Model-Based Anomaly Detection Approach for Analyzing Streaming Aircraft Engine Measurement Data; GT2014-27172; American Society of Mechanical Engineers: New York, NY, USA, 2015.
12. Candelieri, A. Clustering and support vector regression for water demand forecasting and anomaly detection. *Water* **2017**, *9*, 224. [[CrossRef](#)]
13. Candelieri, A.; Soldi, D.; Archetti, F. Short-term forecasting of hourly water consumption by using automatic metering readers data. *Procedia Eng.* **2015**, *119*, 844–853. [[CrossRef](#)]
14. Li, Z.X.; Zhang, F.M.; Zhang, X.F.; Yang, S.M. Research on Feature Dimension Reduction Method for Multivariate Time Series. *J Chin. Comput. Syst.* **2001**, *20*, 565–579. (In Chinese)
15. Xu, Y.; Hou, X.; Li, S.; Cui, J. Anomaly detection of multivariate time series based on Riemannian manifolds. *Computer Science. Int. J. Biomed. Eng.* **2015**, *32*, 542–547.
16. Qiu, T.; Zhao, A.; Xia, F.; Si, W.; Wu, D.O.; Qiu, T.; Zhao, A.; Xia, F.; Si, W.; Wu, D.O. ROSE: Robustness Strategy for Scale-Free Wireless Sensor Networks. *IEEE/ACM Trans. Netw.* **2017**, *25*, 2944–2959. [[CrossRef](#)]
17. Majhi, S.K.; Dhal, S.K. A Study on Security Vulnerability on Cloud Platforms. *Procedia Comput. Sci.* **2016**, *78*, 55–60. [[CrossRef](#)]
18. Qiu, T.; Zheng, K.; Han, M.; Chen, C.P.; Xu, M. A Data-Emergency-Aware Scheduling Scheme for Internet of Things in Smart Cities. *IEEE Trans. Ind. Inf.* **2018**, *14*, 2042–2051. [[CrossRef](#)]
19. Hawkins, J.; Ahmad, S.; Dubinsky, D. HTM Cortical Learning Algorithms. Available online: [https://numenta.org/resources/HTM\\_CorticalLearningAlgorithms.pdf](https://numenta.org/resources/HTM_CorticalLearningAlgorithms.pdf) (accessed on 12 September 2011).
20. Padilla, D.E.; Brinkworth, R.; McDonnell, M.D. Performance of a hierarchical temporal memory network in noisy sequence learning. In Proceedings of the IEEE International Conference on Computational Intelligence and Cybernetics, Yogyakarta, Indonesia, 3–4 December 2013; pp. 45–51.
21. Karagiannidis, G.K.; Lioumpas, A.S. An improved approximation for the Gaussian Q-function. *IEEE Commun. Lett.* **2007**, *11*, 644–646. [[CrossRef](#)]
22. Cocu, A.; Craciun, M.V.; Cocu, B. Learning the Structure of Bayesian Network from Small Amount of Data. *Ann. Dunarea de Jos Univ. Galati Fascicle III Electrotechn. Electron. Autom. Control Inform.* **2009**, *32*, 12–16.
23. Taheri, S.; Mammadov, M. Learning the naive Bayes classifier with optimization models. *Int. J. Appl. Math. Comput. Sci.* **2013**, *23*, 787–795. [[CrossRef](#)]
24. Htm.java. Available online: <https://github.com/numenta/htm.java> (accessed on 9 October 2018).



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).