Anomaly Noise Filtering with Logistic Regression and a New Method for Time Series Trend Computation for Monitoring Systems

Qing Gao, Limin Zhu, Yuxin Lin, Xun Chen

ANT FINANCIAL SERVICES GROUP

Abstract—**Anomaly detection has always been a hot topic in signal processing and machine learning. Convolutional Neural Network (CNN) is an effective technique to detect anomaly. However, at Ant Financial, a simple CNN neglects certain patterns in real-world data that may lead to triggering of false alarms. To reduce the possibility of a false alarm, we run an anomaly noise filtering model after the CNN. In this paper, we introduce techniques to develop the model and a new method of time series trend computation. The model helps increase the accuracy in detecting false anomalies of a rise-fall pattern in the traffic(y-value) of a time series dataset. At the end of the paper, we will present the benchmarks of using our method on real online systems at Ant Financial.**

Index Terms—**noise filtering model, detection algorithms, anomaly detection, rise-fall model**

I. Introduction

Anomaly detection is a common practice in signal processing and is widely applicable in the fields of machine learning. It focuses on the techniques of detecting unexpected sudden drops or rises, trend changes to the traffic of time series data.

At ANT FINANCIAL SERVICES GROUP, anomaly detection methods are used to monitor a large number of indexes (Quantitative Data) of systems. These indexes are tightly correlated and constitute a large network, which reflects the stability of the system operation. For example, by tracking the call volume of a particular system, emergency response teams can respond quickly to alerts generated by system exceptions or external attacks. The accuracy of alarms is important due to the cost of human intervention in debugging the validity of alarms.

In practical work, anomaly detection is generally categorized as a sudden positive or negative spike in data traffic. In most cases, these two events happen abruptly, however that is not entirely the case. In some situations, the traffic rises and falls back after an arbitrary prolonged time period. Let us refer this phenomenon as the "rise-fall" pattern in this paper.

Figure 1 illustrates some examples of this phenomenon, where the solid curve represents data of "today" (date of data snapshot), and other curves are past data in regard to today. From the figure, we can see the rise-fall pattern in today's curve will trigger a false alarm with previous methods of anomaly detection. From our observation, the rise-fall pattern is usually not considered as an anomaly, particularly when the traffic rises and falls back to its previous trend, as this kind of data fluctuation is often triggered by business promotional activities.

Causes of this phenomenon varies. For example, the rise-fall pattern of the trade volume of Tmall (a popular Chinese online retailer shop) could be caused by "Miao Sha" (the name of a promotional feature used in the Tmall application). In terms of interface monitoring, this trend can be caused by a sudden surge and drop of external calls. With millions of indexes in our system, the sheer number of indexes makes it impossible to track down the root causes.

In a real-world setting, every alarm should be validated. Too many false alarms can cause wasted resource, and decline in user faith. That is why it's necessary to explore methods to identify false alarms from the "rise-fall" pattern.

In this paper, we propose a method to identify false alarms from the rise-fall pattern. The main points of this paper are summarized below:

1) We propose a new problem, the rise-fall event which occurs in a real-world scenario with actual data for references.

2) A new noise filtering model ("rise-fall model") to filter out possible false alarms.

3) A new method to compute time series trend, which we will refer to as "Trend Decomposition based on First-order Difference (TDFD)"in our paper, and the decomposed trend is named as "baseline".

Disclaimer: Because our filtering model is highly customized for detecting the falling edge of an anomaly pattern, it may only work specifically for a rise-fall pattern and does not work for a "fall-rise" pattern. We are open to suggestions and possible work in the future that may improve upon our methods.

In this paper, Section 2 reviews related works. Section 3 presents detailed model and data sets. Section 4 compares the results of an anomaly detection with and without the rise-fall model. Section 5 is conclusion.

II. Related Works

According to our knowledge, at the time of writing this paper, there has been no related paper published on works of this matter. However, our method falls into the category of time series classification, therefore some previous methods of time series classification methods can be substituted in place of our method.

Methods for classifying time series can be generally divided into two categories, supervised learning (semi-supervised learning) and unsupervised-learning. We can place semi and fully supervised learning as one category since they both allow the usage of labelled examples. A plethora of research has been done in the field of supervised classification methods. Hand-crafted features are extracted and then fed into classification models (logistic regression, decision trees, boosting trees), such as Bag-of-Words (BoW) [1], Bag-of-features (TSBF) [2], Bag-of-SFA-Symbols (BOSS) [3], BOSSVS [4]. Another trend in recent years is using neural networks to automatically extract features from series to represent their patterns. Ref. [5] puts forward a baseline of deep convolutional networks and Ref. [6] combines LSTM with convolutional networks to boost performance. Semi-supervised learning can be divided into five classes, which are low density separation, graph-based methods, co-training methods, and self-training methods as given by [7, 8].

Unsupervised techniques are usually statistically driven methods, including sliding windows threshold, outlier tests such as ESD and k-sigma, changepoint detection. Another branch of unsupervised learning is model based, such as isolation forest [9], variational auto-encoders [10, 11].

III. Model And Data Set

In this section, we will introduce the dataset in detail and then propose our model of detecting rise-fall patterns and its feature extraction methods.

A. Data Set

We first randomly select 13660 samples of time-series data from past datasets, with 5415 anomalies and 8245 non-anomalies. We collected an annual dataset of all anomalies in our system, including real and false alarms. For each sample anomaly, 5 sets of historic data are snapshotted for reference: past 24 hours, past 48 hours, past 4 days, 8, and 15 days ago (For simplicity we will refer these datasets as "T minus 1, 2, 4, 8 and 15" respectively). For T-1, the data is from 00:00 to the current minute in time (N) of the snapshot is used. For all other datasets, the data from 00:00 to the next hour of N - 1 minute. Commonly in practice, when the traffic falls down after keeping steady for 1 hour, it's very likely to be anomaly, so we focus on the data within a window of 60 minutes. According to experience the models are optimized under this assumption. There are samples with "rise-fall" pattern wider than 60 minutes, which will be considered in future study. The reference data is extracted from 1,4,8,15 days ago because our data traffic usually shows a stable 7-day periodic pattern. Thus, these four days' data can reveal most of its implicit pattern. Reference dates and window sizes can be chosen differently to suit varying data traffic patterns. For same analysis with other data set of non-periodic samples, the reference days should be chosen carefully.

Figure 1 shows several different rise-fall patterns. Traffic could suddenly rise up and down, forming a spike, or rise and drop after a long time. Even the slope of the traffic can be different, traffic may rise or drop suddenly or gradually. From the figure, we can see that for most samples, the traffic trends are not flat, but can be either positively or negatively sloped which further increases the difficulty of determining an anomaly.

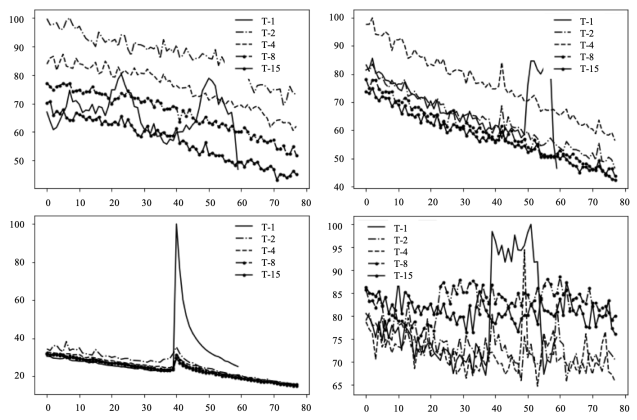


Fig. 1. The types of rise-fall patterns, where x-axis is the time series starting from 0, and y-axis is the real value of data, solid curve is the data of T-1 with totally 60 minutes, other curves are reference data of the same period respectively of historical T-2, T-4, T-8 and T-15 data with 78 minutes in all, consisting of 60 minutes before the current minute and 18 minutes after.

Figure 1 also shows that the distribution of traffic differs among samples. Sometimes the traffic has consistent patterns at certain intervals, such as the up two plots, while other times the traffic does not conform to a strict pattern. Also note that the magnitude of T-1 traffic differs from historical datasets. For example, when promotional activities happen, T-1's traffic on average will be higher than other historical datasets, as shown by the bottom-left plot.

As we are using our model to compute sample datasets, we noticed our model's performance was not consistent among different samples. We decided to investigate more and find out what properties of a sample affected our model's performance. We decided to split the samples into four categories with two features:

1, Similarity ρ: the median value of Pearson correlation coefficients among our historic datasets

2, Difference δ: the average of T-1's traffic minus the average of historical traffic

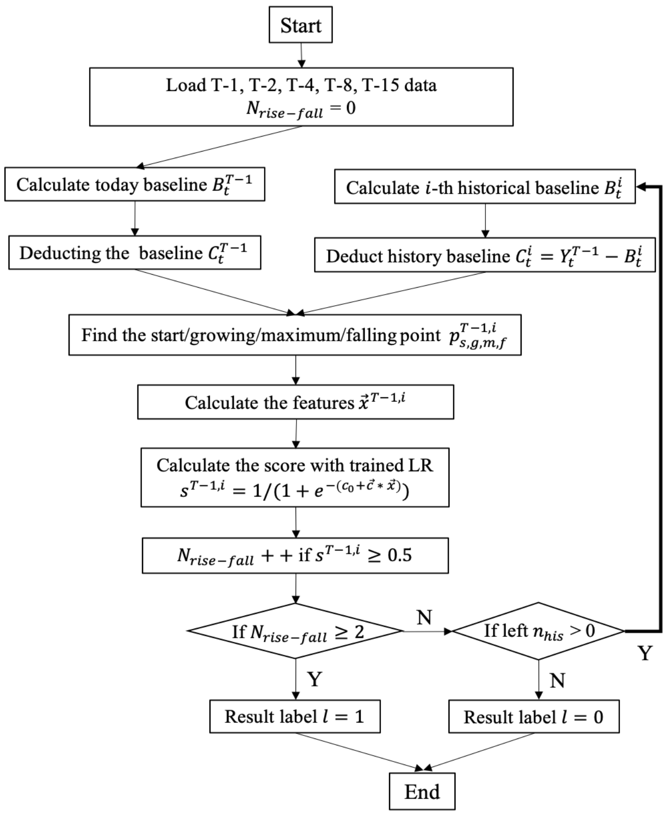
The four categories are:

1. Periodic & Non-Uptrend: $\rho \geq 0.95$ & $\delta \leq 0$;
2. Non-Periodic & Non-Uptrend: $\rho < 0.95$ & $\delta \leq 0$;
3. Periodic & Uptrend: $\rho \geq 0.95$ & $\delta > 0$;
4. Non-Periodic & Uptrend: $\rho < 0.95$ & $\delta > 0$.

B. Model

The rise-fall pattern has a rising stage and a falling stage. If there is a valid anomaly, the traffic should fall below its previous trend to a relatively large degree. Imagine a scenario where the trade volume of Tmall rises and drops down after 24 hours during a promotion. If the trade volume drops significantly below the its previous volume 24 hours ago, something must have gone wrong with the promotion, causing a reduction in overall sales which must be alerted. Now say if the trade volume drops but is relatively the same with the previous trade volume 24 hours ago, then it must mean the promotion went smoothly and didn't cause an overall reduction in sales. This type of rise and fall pattern should not be alerted and our filtering model should ignore these types of anomalies.

The sample of second plot in Fig. 1 is an example of a valid anomaly, where the traffic falls below its previous trend significantly. Samples of other plots in Fig. 1 are all non-anomalies, where the traffic falls close to the previous trend but not below it no matter how the traffic changes.



Start
Load T-1, T-2, T-4, T-8, T-15 data
$N_{rise-fall} = 0$
Calculate today baseline B_t^{T-1}
Calculate i-th historical baseline B_t^i
Deducting the baseline C_t^{T-1}
Deduct history baseline $C_t^i = Y_t^{T-1} - B_t^i$
Find the start/growing/maximum/falling point $p_{s,g,m,f}^{T-1,i}$
Calculate the features $\vec{x}^{T-1,i}$
Calculate the score with trained LR
$s^{T-1,i} = 1/(1 + e^{-(c_0 + \vec{c} * \vec{x})})$
$N_{rise-fall}$ ++ if $s^{T-1,i} \geq 0.5$
If $N_{rise-fall} \geq 2$
N
If left $n_{his} > 0$
Y
Y
N
Result label $l = 1$
Result label $l = 0$
End

Fig. 2. the rise-fall model, including TDFD and logistic regression.

Therefore, we propose a new method of decomposing time series and a linear regression model using features extracted from decomposed time series. The model focuses on the anomalies that have been triggered within an hour but with at least 24 hours of historical data as reference. The flow chart of the model is as shown in Fig. 2 and five main steps are:

a) Calculate the decomposed trendline (which we will call as "baseline") based on TDFD
b) Find the start, growing, maximum and falling point for the curve after deducting the baseline
c) Calculate the features of the new curve with the four points
d) Calculate the score of rise-fall with a logistic regression model [12]
e) Decide the result for each event

1) Calculating the Baseline

T-1 traffic data is a time series $Y = \{y_1, y_2, \dots, y_T\}$ with time length T, which is regarded as the combination of baseline and temporary component:

$$Y = B + C \tag{1}$$

where the baseline part $B = b_{1:T}$ represents the long-term trend, and the temporary component $C = c_{1:T}$ represents the short-term fluctuation. The first step is to calculate the baseline B_t of traffic data. The rise-fall event often happens when the traffic is stable in past historical datasets. As shown in Fig. 3 first plot, the solid and dashed-dot curves are T-1 traffic and a past dataset, and the other two curves are their baselines. The baseline is determined with two-stage loops in **Algorithm 1**.

Algorithm 1

```
Input: $\boldsymbol{Y} = \boldsymbol{y}_{1:T}$ is the original data
Output: $B = b_{1:T}$ is the baseline of $Y$
Initialization:
Get moving average window $w$ with the smallest moving $\sigma_w^Y$
    $w = argmin(\sigma_w^Y) = argmin\left(\frac{\sum_{i=1:T-w}\sqrt{\frac{\sum_{t=i:i+w-1}(y_t - \overline{y_{i:i+w-1}})^2}{w}}}{T-w}\right)$
        where $w \in [2, 8]$                                    (i)
Split the original series $y_{1:T}$ into $n$ equal parts $\{Y^{(i)}\}, i = 1{:}n$
    $Y^{(i)} = y_{(i-1)\times T'+1:i\times T'}, \quad T' = \frac{T}{n}$
with $n_{T-1} = 1$ and $n_{historical} = 4$ now
Loop 1: for $i = 1{:}n$ do
  Step a. Moving average of series as $MY^{(i)} = Y_w^{(i)}$
    The first order difference
      $dY_w^{(i)} = y_{t,w}^{(i)} - y_{t-1,w}^{(i)}, t = (i-1)\times T' + 2{:}i \times T'$
  Step b. Moving standard deviation $\sigma_w^{MY^{(i)}}$ of $MY^{(i)}$ with Eq(i)
    Split $dY_w^{(i)}$ and $MY^{(i)}$ into $m$ continuous segments $s_{1:m}$
      $s_j = \{dY_{j,w}^{(i)}, MY_j^{(i)}, t_j^{(i)}\} = \left\{dy_{t_j^{(i)},w}^{(i)}, y_{t_j^{(i)},w}^{(i)}, t_j^{(i)}\right\},$
      $t_j^{(i)} = (i-1)\times T' + (T'_{j-1}{:}T'_j)$
where $s_j$ is continuous in time, $\max(dY_w^{(i)}) - \min(dY_w^{(i)}) < \sigma_w^{MY^{(i)}}$
    Sort the segments with $\overline{dY_w^{(i)}}$ and length of segment.
  Step c. $\varepsilon_{i,0} = Inf, \vec{\alpha} = \{0,0,0\}, t_{core}^{(i)} = t_{cand}^{(i)} = \emptyset, MY_{core}^{(i)} = MY_{cand}^{(i)} = \emptyset$
    Loop 2: for sorted segments $j = 1{:}m$ do
        $t_{cand}^{(i)} = \{t_{core}^{(i)}, t_j^{(i)}\}, y_{cand}^{(i)} = \left\{y_{core}^{(i)}, y_{t_j^{(i)},w}^{(i)}\right\}$
      Fit the segment
        $y_{cand}^{(i)} = \alpha_0 + \alpha_1 \times t_{cand}^{(i)} + \alpha_2 \times {t_{cand}^{(i)}}^2 + \varepsilon_t$
      Predict values
        ${y_{cand}^{(i)}}' = y_{cand}^{(i)} - \varepsilon_t = \alpha_0 + \alpha_1 \times t_{cand}^{(i)} + \alpha_2 \times {t_{cand}^{(i)}}^2$
      The relative error
        $\varepsilon_{i,j} = \sqrt{\sum_t \left(y_{cand}^{(i)} - {y_{cand}^{(i)}}'\right)^2}$
      If $\varepsilon_{i,j} < \varepsilon_{i,0}$ then
        $t_{now}^{(i)} = t_{cand}^{(i)}, y_{now}^{(i)} = y_{cand}^{(i)}, \varepsilon_{i,0} = \varepsilon_{i,j}, \vec{\alpha} = \{\alpha_0, \alpha_1, \alpha_2\}$
  Step d. Predict values in whole part
      $Y_w^{(i)'} = \alpha_0 + \alpha_1 \times t + \alpha_2 \times t^2, \ t = (i-1)\times T' + 1{:}i \times T'$
Step e. Combine predict values $b_{1:T} = \left\{Y_w^{(i:n)'}\right\}$
```

First, the line is split into segments by the first order difference and the longest segment with the smallest first order difference is selected. Second, the selected segment is used as a kernel to expand to the whole line by fitting with a second order polynomial model. Then, the segment with the larger difference is joined with the selected segment and fitted again. If the relative error between the original line and predicted curve drops, the newly joined segment combined with the original selected segment is set as the new kernel for expansion. Repeat above two steps until all segments are checked and the baseline is determined.

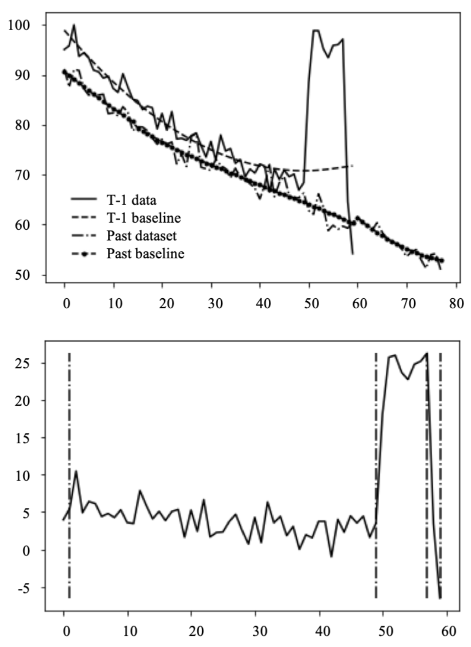


Fig. 3. The process of baseline calculation, where x-axis is the time series starting from 0, and y-axis is the data traffic. In the first image, solid curve is T-1 data, dashed-dot curve is a past dataset, the dashed and dashed-dot-dashed lines are their baselines. In the second image, the solid curve is data of T-1 after deducting the baseline of its reference data, and the dashed-dot lines are the places of start, growing, maximum and falling points.

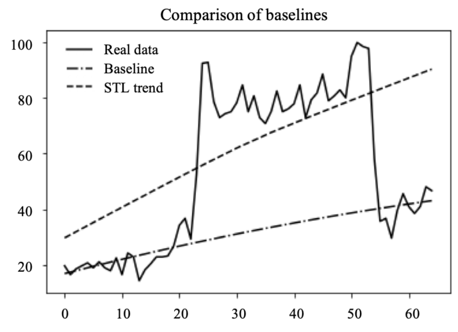


Fig. 4. Comparison between the trends calculated by the baseline method in this paper and traditional STL method, where solid curve is the original time series, dashed-dot line is the baseline for data, and the dashed line is the trendline computed using STL.

Compared to the traditional method Seasonal-Trend decomposition procedure based on Loess (STL, [13]) of calculating the trend of a time series, the TDFD here is better in rise-fall situations, as shown in Fig. 4. In this figure, an obvious rise-fall with growing trend is shown. The STL method can only extract the growing trend influenced by the peak data, while the TDFD catches the trend successfully and leaves a complete peak. Traditional trend decomposition methods based on iteration and loess are all affected by the peak of the data and the trend would be similar with STL here.

2) Find the Start, Growing, Maximum and Falling Point for the Curve after Deducting the Baseline

According to equation (1) and the baseline determined in the above section, we can get the temporary component $C = c_{1:T} = y_{1:T} - b_{1:T}$, as shown in the second plot of Fig. 3. In the figure, the solid curve is T-1's traffic after deducting the baseline of its reference data. The baseline of reference data is used to replace the baseline of T-1 to get a new curve C without a trendline. To describe the shape of C, four main points of the new curve are extracted. The four vertical dotted lines mark the start, growing, maximum and falling point, $p_{s,g,m,f}$, respectively. The four points help locate the position of the peak of the rise-fall event. Given a specific window defined by users, the growing point is at the rising edge of the latest rise-fall event. The maximum point is the local maximum of the latest rise-fall pattern. For the traffic without a rise-fall pattern, the maximum point is the global maximum of the whole window. The falling point is at the falling edge of the latest rise-fall event. Now with the first 3 points, we know when the rise-fall event starts and ends. The remaining part of the curve minus rise-fall event follows the baseline, the start point should be where the remaining part of the curve started to stabilize. For windows with two or more rise-fall patterns, the start point is after the falling point of the second to last rise-fall pattern.

Following the order of $p_f \to p_m \to p_s \to p_g$ by looping the values of $c_{1:T}$ in order of $T \to 1$, we get the four points:
Initialization: $C = c_{1:T}$, moving average window $w = w_{today}$, Fluctuation $\sigma = \sqrt{\frac{\left(\sigma_w^{C^2} + \sigma_{DMY}^{refer^2}\right)}{2}}$, where $\sigma_w^{C^2}$ is the moving standard deviation of T-1 C, σ_{DMY}^{refer} is the standard deviation of first order difference of historical datasets and zero of T-1.

Falling point p_f
```
for  $t = T, T-1, \dots, 1$  do
    if  $c_t < c_{t-1}$
        $p_f = t$
        end
```

Maximum point p_m
```
for  $t = p_f - 1, p_f - 2, \dots, 1$  do
    if  $c_t > c_{t-1}$
        $p_m = t$
    else if  $c_{p_m} - c_t \geq 2 \times \sigma$  and  $c_{t-1} - c_t < \sigma$
        end
```

Start point p_s
```
for  $t = p_m - 1, p_m - 2, \dots, 1$  do
    $\bar{c}_t = \frac{\sum_{t'} c_{t:p_m}}{p_m - t}$
    if  $c_t < c_{t+1}$
        $p_s = t$
    else if  $c_t - \bar{c}_t \geq 2 \times \sigma$  and  $c_{p_s} - c_{p_f} > 2 \times \sigma$
        end
```

Growing point p_g
```
for  $t = p_s + 1, \dots, p_m - 1$  do
    $\bar{c}_t = \frac{\sum_{t'} c_{p_s:t}}{t - p_s}$
    if  $c_{t-1} > c_t$  and  $c_{t+1} - c_t > 2 \times \sigma$
        continue
    if  $c_t \leq \bar{c}_t$
        $p_g = t$
    else if  $c_t - c_{p_s} > 2 \times \sigma$
        end
```

If the falling value does not fall below any point, the peak is recognized as a rise-fall pattern and a possible false alarm could be intercepted.

3) Calculate the Features of the New Curve with the Four Points

To describe the shape of the peak, we calculate eight features $\vec{x}$ here with the four points and the values of the curve. To alleviate the influence caused by different scales of data, we use the below formula to normalize each feature,

$$t(x_1, x_2) = \frac{x_1 - x_2}{\sigma_w^C} \tag{2}$$

where the fluctuation of the curve σ_w^C is determined by the temporary change part C_t. The input and feature definitions are described in Table I.

TABLE I
FEATURES OF TEMPORARY COMPONENT

Features	Definition	Parameters
Down Zero	$t\left(c_{p_f}, 0\right)$	0.1251
down	$t\left(c_{p_f}, c_{p_g}\right)$	0.4754
Before mean	$t\left(c_{p_f}, \overline{c_{p_s:p_g}}\right)$	0.3200
In	$t\left(\overline{c_{p_g:p_f}}, \overline{c_{p_s:p_g}}\right)$	0.0006
Rise before	$t\left(c_{p_g:p_m}, \overline{c_{p_s:p_g}}\right)$	-0.3106
Now	$t\left(c_T, c_{p_g}\right)$	-0.3252
Now before min	$t\left(c_T, \min c_{p_s:p_g}\right)$	0.0006
Max before mean	$t\left(c_{p_m}, \overline{c_{p_s:p_g}}\right)$	0.1735
Constant	1.0	1.5878

The eight features $\vec{x}$ of the temporary component C. The parameters and the constant are weights of the trained logistic regression model.

4) *Get the Score of Rise-fall*

After the features are obtained, the score of rise-fall can be determined by the trained logistic regression model,

$$score = \frac{1}{1+e^{-(c_0+\vec{c}\cdot\vec{x})}} \tag{3}$$

where $\vec{x}$ is the vector of eight features, $\vec{c}$ and c_0 are the corresponding parameters and constant. The trained parameters are listed in TABLE I. We use 0.5 as the threshold for the scenario of rise-fall existing or not as usual.

5) *Get the Result for Each Sample*

For each sample, we will compute the temporary component of T-1's traffic by subtracting the baseline of both T-1 and historical days from T-1's original data. This will create five time series. For each of these series, we will extract its features according to (4) and compute its abnormal score. The reason being that with comparison to different covariates, we can increase the robustness of our model. Therefore, a sample is only labelled as a rise-fall event when at least two out of five scores are above 0.5 ($N_{rise-fall} \geq 2$). Detailed conditions are listed follow.

```
Decisions
Initialization:
        $N_{rise-fall} = 0$,    $res_{rise-fall} = false$
Loop: for $i = T-1, 2, 4, 8, 15$
    if $score \geq 0.5$
        $N_{rise-fall}$++
Decision: if $N_{rise-fall} \geq 2$
        $res_{rise-fall} = true$
End
```

C. *Measure the Performance of Model*

The results of the confusion matrix are listed in Table II. For anomaly samples, it would be marked as lose (FN) if rise-fall method intercepts and right (TP) otherwise. For non-anomaly samples, if it is intercepted then the mark is not (TN) otherwise is wrong (FP). To measure the effect of models, precision, recall, accuracy, f1 score are quoted, and the definitions of each are listed in TABLE II.

TABLE II
PERFORMANCE OF MODEL

	Anomalies	Non-anomalies
Not intercept	TP	FP
Intercept	FN	TN
Precision	$p = TP/(TP+FP)$	
Recall	$r = TP/(TP+FN)$	
Accuracy	$accu = \dfrac{TP+TN}{TP+FP+FN+TN}$	
F1 score	$f1 = \dfrac{2}{\frac{1}{p}+\frac{1}{r}}$	

The confusion matrix and precision, recalling, accuracy, f1 score of the model.

IV. RESULTS

A. *Results of Anomaly Detection Without and With Rise-fall Model*

The results of anomaly detection with and without the rise-fall model are listed in TABLE III. First, we use a convolutional neural network model (CNN, [5]) to detect the falling-edge of the traffic. The table shows that almost all valid anomalies are recognized by CNN model, with the recall being 0.9976. This model has a low precision of only 0.6431. In a real application, this model is only good enough to detect possible anomaly declines with many false alarms. The occurrence of rise-fall pattern is small relative to the sample size, thus using regular business traffic is not enough to optimize CNN model with perfect precision. Supposedly if every alarm was responded to by a human, the model would have wasted a lot of human resources due to over one-third of the alarms being invalid. Now with the rise-fall model attached, the number of false alarms is significantly reduced, and the precision increases to 0.8630. Although in some cases valid anomalies may be wrongly intercepted, but the decrease in recall cannot be comparable to the increase of precision, leading to higher accuracy and f1 score. The intercepted samples are probability not sufficiently abnormal so that users would not care about. In reality, anomaly detection as well as rise-fall detection is processed every minute, thus recall rate in the production environment will be higher.

TABLE III
RESULTS WITH AND WITHOUT RISE-FALL MODEL

	Without	With
Confusion matrix	5246, 2998 13, 5402	7475, 769 572, 4843
Precision	0.6431	0.8630
Recall	0.9976	0.8944
Accuracy	0.7796	0.9018
F1 score	0.7820	0.8784

The confusion matrix and precision, recalling, accuracy, f1 score of detections without and with rise-fall model.

B. Results in Detail

TABLE IV
RESULTS OF RISE-FALL MODEL IN DIFFERENT DATA TYPE

		Periodic, Non-Uptrend	Non-Periodic, Non-Uptrend	Periodic, Uptrend	Non-Periodic, Uptrend
Without	Precision	0.7709	0.6798	0.3559	0.1720
	Recall	0.9973	0.9992	1.0	1.0
	Accuracy	0.8470	0.8447	0.5847	0.3932
	F1 score	0.8696	0.8092	0.5250	0.2936
With	Precision	0.9010	0.8807	0.6650	0.5115
	Recall	0.9027	0.8877	0.8656	0.7708
	Accuracy	0.8995	0.9234	0.8691	0.8783
	F1 score	0.9018	0.8842	0.7521	0.6150

The precision, recalling, accuracy, f1 score of detections without and with rise-fall model in detail.

Previous results are the average of all samples, while the distribution of real traffic differs a lot. One method may perform well in one situation but bad for others. The result listed in TABLE IV shows the difference in performance of the anomaly detection with and without the rise-fall model when processing traffic data of different types. For all types of data, the precision and accuracy with rise-fall model is better than those without rise-fall model. This is consistent across all average situations. The precisions of both with and without rise-fall model decrease with the increase in the complexity of data. This can be seen from the drops in precision below 70% when the average value of T-1 is significantly larger than that of past historical datasets (regarded as promotional activities).

C. Summary

The theory, condition and range in application, and performance of the rise-fall model have been described in detail in previous sections. To detect rise-fall pattern, at least one reference day is required. The rise-fall model can intercept most false alarms caused by anomalies and the precision is significantly increased from 64.31% to 86.30% with a slight decrease in recall. Meanwhile, the rise-fall model proposes a new algorithm to calculate the trendline, which produces more accurate trendlines in rise-fall situation than the traditional STL model. However, there still remains room for improvement in some cases where the trend of historical data is not consistent with that of T-1 data, which will result in inaccurate trendline computation.

V. CONCLUSION

In this paper, we proposed a new problem for monitoring systems, the rise-fall pattern. Different from sudden traffic spikes or drops, the rise-fall pattern also includes gradual rise and drop patterns in data traffic. This type of pattern can easily be misinterpreted as anomalies by previous anomaly detection models. Since the rise-fall pattern have not been carefully scrutinized before in traditional anomaly detection models, we propose a new dataset and a baseline model to handle this type of traffic pattern. A new way of decomposing time series has been used in our model and shown its effectiveness when compared with traditional STL method. Experiments show that the accuracy and overall performance (F1) has increased significantly when rise-fall model is combined with our anomaly detection model. When inconsistencies exist among recent and historical datasets and when complexity of data increases, our model still shows room for improvement, which directs our future work.

REFERENCES

[1] J. Lin, E. Keogh, L. Wei, and S. Lonardi, Experiencing sax: a novel symbolic representation of time series, Data Mining and knowledge discovery, vol. 15, 2: 107–144, 2007.
[2] M. G. Baydogan, G. Runger, and E. Tuv, A bag-of-features framework to classify time series, IEEE transactions on pattern analysis and machine intelligence, vol. 35, 11: 2796–2802, 2013.
[3] P. Schäfer, “The boss is concerned with time series classification in the presence of noise,” Data Mining and Knowledge Discovery, vol. 29, 6: 1505–1530, 2015.
[4] P. Schafer, Scalable time series classification, Data Mining and Knowledge Discovery, 1–26, 2015.
[5] Z. Wang, W. Yan, T. Oates, Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline[J]. 2016.
[6] F. Karim, S. Majumdar, H. Darabi, et al. LSTM Fully Convolutional Networks for Time Series Classification[J]. IEEE Access, 6(99):1662-1669, 2017.
[7] O. Chapelle, B. Scholkopf, A. Zien, Semi-Supervised Learning. In press. MIT Press., 2006.
[8] X. Zhu, Semi-supervised learning literature survey. Technical report, no. 1530, Computer Sciences, University of Wisconsin-Madison, 2005.
[9] F. Liu, K. Ting, Z. Zhou. Isolation-based anomaly detection, ACM Transactions on Knowledge Discovery from Data (TKDD) 6.1, 3, 2012.
[10] D. Park, Y. Hoshi, and C. C. Kemp, “A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-based Variational Autoencoder,” CoRR, vol. abs/1711.00614, 2017.
[11] H. Xu, W. Chen, N. Zhao, et. al., Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Application, arXiv: 1802.03903 [cs.LG], doi: 10.1145/3178876.3185996.
[12] H. F. Yu, F. L. Huang, C. J. Lin, Dual coordinate descent methods for logistic regression and maximum entropy models. Machine Learning 85(1-2): 41-75, 2011.
[13] B. Robert, S. William, I. Terpenning, STL: A seasonal-trend decomposition procedure based on loess. Journal of Official Statistics 6.1, 3, 1990.