One-step Predictive Encoder - Gaussian Segment Model for Time Series Anomaly Detection

Jiachen Zhao*, Yongling Li†, Haibo He‡, and Fang Deng*
*Department of Automation
Beijing Institute of Technology, Beijing, China,100081
Email: zhao_jiachen@163.com, dengfang@bit.edu.cn
† State Key Laboratory of Rail Traffic Control and Safety
Beijing Jiaotong University, Beijing, China,100041
Email: liyongling@bjtu.edu.cn
‡ Department of Electrical, Computer and Biomedical Engineering
University of Rhode Island, Kingston, RI 02881
Email: he@ele.uri.edu

***Abstract*—Unsupervised anomaly detection for time series is of great importance for various applications, such as Web monitoring, medical monitoring, and device fault diagnosis. Time series anomaly detection (TSAD) aims to find the observations that most different from others in a sequence of observations. With the development of deep learning, deep-autoencoder-based methods achieve state-of-the-art performance. These methods are usually able to find single anomaly points but fail to detect the anomaly segment and the change point. To tackle this problem, this paper proposes a novel TSAD method, which consists of a bidirectional LSTM (BiLSTM) autoencoder and a subsequent Gaussian segmentation model. BiLSTM encodes a time series in a predictive format from both positive and negative time directions, then outputs the latent feature vectors and restructured errors. After that, the latent features are used to find anomaly segments by the Gaussian segment model; the restructured errors are used to find change points and extreme single anomaly by a scoring function. In this way, our method can find all three kinds of anomaly points. Experiments on two real-world datasets demonstrate the effectiveness of the proposed method.**

***Index Terms*—Time series, anomaly detection, change point detection, deep learning.**

I. Introduction

Detecting unexpected events or rare items in the temporal evolution of a system is of importance in both fundamental machine learning research and industrial applications. This task is generally referred to as time series anomaly detection, which aims to find the data points that most differ from other observations in a temporal sequence of observations. The time series anomaly detection technique has a wide range of applications such as Web attack detection, medical monitoring [1], and device fault diagnosis. For example, Microsoft also builds a time series anomaly detection service [2] to monitor various web metrics (such as Page Views and Revenue), which further help engineers move faster in solving live site issues.

Over the years, a considerable amount of literature studies on the TSAD problem. Most traditional statistical methods are based on similarity search [3], regression [4], or clustering [5, 6]. Since these methods typically have assumptions for the time series, it needs expert's knowledge to extract features and to build a suitable detector for a given time series. Recently, neural-network-based autoencoders draw researchers' attention because of their good performance in the TSAD problem. The philosophy of the autoencoder-based TSAD method is to compress the original sequence into a fixed-length hidden representation and then reconstruct the input sequence based on the hidden representation. Since the hidden representations are compact, only samples with common patterns can be well reproduced and the outliers with specifics patterns will be less reproduced. Therefore, the reconstruction error can be used as the measure of the anomaly. Hawkins et al. [7] proposed a so-called replicator neural network for outlier detection, which is essentially a fully-connected autoencoder. Tung et al. [8] merged several sparsely-connected recurrent neural networks into an ensemble framework to advance TSAD performance. Ergen et al. [9] combined the LSTM neural network and one-class SVM into an end-to-end structure to detect the anomaly points. However, all these methods only focus on extreme anomaly points, not considering the change points and anomaly segments, as defined in III-A

Despite the rapid development of TSAD methods, there are still some open challenges in time series anomaly detection.

(1) **Lack of Labels.** A time-series often has more than thousands of observations along the time axis, but the abnormal points are usually very sparse and their happening time is random. What's worse, time-series data is more abstract than image data. As a result, there is no easy way to label enough data to train a supervised classifier. The supervised model also suffers the unbalanced problem [10].

(2) **Hard to accurately locate.** Some existing methods [8, 9] segment the time series with a sliding window, treat each window of observations as an independent sample and finally detect whether each sample is an anomaly or not. As a result, these methods fail to detect whether a single observation is

This work was supported in part by Key Program of National Natural Science Foundation of China (#61933002) and china scholarship council.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

an anomaly or not. However, we usually hope an anomaly detection method can find the exact time point of each anomaly rather than finding a sub-sequence that contains the anomaly.

(3) **Multiple kinds of anomaly points**. Autoencoder-based TSAD methods perform well in detecting the extreme anomaly points, but if there is a successive sequence of anomaly points in the time series, these methods may encode the anomaly segment as normal observations. So an anomaly detector is supposed to find all kinds of anomaly points.

Considering the above challenges, we propose a novel TSAD method, referred as One-step predictive encoder - gaussian segment model (OPE-GSM). The OPE is based on the BiLSTM autoencoder, different regular encoder OPE reconstructs the time series in a predictive form from bidirections. Based on the restructure errors, a scoring function is used to detect the extreme anomalies. To detect the change points and anomaly segments, a Gaussian segment model is proposed, which divides the latent features into segments by maximizing a cumulative gaussian likelihood function. To sum up, the contributions of this paper are three-fold as follows,

- We propose a BiLSTM-based one-step predictive encoder (OPE), which can accurately locate the extreme anomalies because of the bidirectional encoding.
- Based on the output of the OPE, we propose a Gaussian segment model to detect the change points and the anomaly segments.
- We conduct experiments on two real-world datasets to evaluate our proposed method, and the results shows that OPE-GSM outperforms the best comparison method with 3.67% improvement.

The rest of this paper is organized as follows. Section II gives preliminaries to the LSTM neural network. Section III details the proposed method. Section IV presents the experimental results and analysis. The conclusion is given in Section V.

II. Preliminaries

Many existing papers [11, 12] confirmed that LSTM can model the correlation of observations in time series. In this section, we give a brief introduction to the classical LSTM neural network. LSTM is a kind of recurrent neural network (RNN), which uses its inherent memory structures (cell states) to store the "time" information and use the control structure (gates) to regulate the amount of stored information. We follow classical structure of LSTM in [13], where the internal propagation equations follow.

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{ci} \circ \boldsymbol{c}_{t-1} + \boldsymbol{b}_i) \tag{1a}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{cf} \circ \boldsymbol{c}_{t-1} + \boldsymbol{b}_f) \tag{1b}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \circ \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \circ tanh(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \tag{1c}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{co} \circ \boldsymbol{c}_{t-1} + \boldsymbol{b}_o) \tag{1d}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \circ tanh(\boldsymbol{c}_t) \tag{1e}$$

where $\boldsymbol{x}_t$ is the input vector, $\boldsymbol{c}_t$ and $\boldsymbol{c}_{t-1}$ are the cell state vectors at time t and $(t-1)$, $\boldsymbol{h}_t$ and $\boldsymbol{h}_{t-1}$ are the output vectors at time t and $(t-1)$. Besides, $\boldsymbol{i}_t$, $\boldsymbol{f}_t$ and $\boldsymbol{o}_t$ are the input, forget and output gates, respectively. Here, σ is the sigmoid function and $\circ$ is the elementwise multiplication. Furthermore, Ws are the weight matrices whose subscripts indicate which two variables are connected by this wight matrix. For example, $\boldsymbol{W}_{xi}$ is the wight matrix from the input vector $\boldsymbol{x}_t$ to the input gate $\boldsymbol{i}_t$. $\boldsymbol{b}$s are biases of LSTM.

In this LSTM structure, the cell state vector $\boldsymbol{c}_t$ stores the hidden features of the input time series, Hense, we can use the LSTM to obtain a fixed-length vector representation for each time series $\boldsymbol{X} = \{\boldsymbol{x}_i\}_1^T$. In the following contents, we also call the cell state vector by latent feature vector and use it to segment the time series as presented in Section III-E. The output of LSTM is also a sequence, so it can be used to restructure the input sequence and the restructured error vector can be used to detect the anomaly points. In this paper, we use a variant of LSTM, the bidirectional LSTM, to restructure the time series and detect the anomaly points, as shown in Section III-C and III-D.

III. Proposed Method

A. Problem formulation

We consider a given time series $\boldsymbol{X} = \{\boldsymbol{x}_i\}_1^T$, where T is the length of the time series, $\boldsymbol{x}_i = [x_{i,1}\ x_{i,2}\ \cdots\ x_{i,d}] \in \mathbb{R}^d$ denotes the i-th observation and $d \in \mathbb{Z}^+$ is the dimension of observations. Following existing papers [14], we assume that the majority of the observations are normal and the anomalous observation randomly happens. We also follow the unsupervised scenario that we have no labeled time series as the training set. Our goal contains two parts: (1) to find a scoring function to measure $\boldsymbol{x}_i$'s anomaly degree, (2) to find a decision function to determine whether $\boldsymbol{x}_i$ is anomalous or not. Throughout the paper, vectors are denoted by boldface lowercase letters and matrices are represented by boldface uppercase letters.

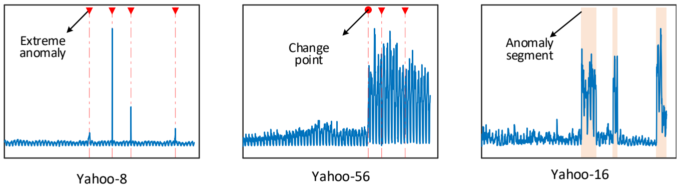


Fig. 1. Examples of anomaly points in real-world datasets. The blue curve is the original time series. The vertical line with the triangle sign indicates the extreme anomaly; the vertical line with the circle sign indicates the change point; the colored background indicates the anomaly segment.

After reviewing the related literature [15], we categorize the anomaly points into three classes: (1) the extreme anomaly, which extremely differs from the temporally neighboring observations; (2) the change point, whose antecedent and subsequent observations follow different latent models (distributions, ARMA models, etc.). (3) the anomaly segment, which contains a successive sequence of anomalous points. Fig.1 shows examples of three types of anomaly points in the **Yahoo** dataset. The proposed method can effectively find all three types of anomaly points.

B. Overall framework

The main idea of our method is to use an encoder-decoder structure to reconstruct the raw time-series data, and then use the restructured errors to predict the extreme anomaly and use the latent compact representation to predict the change points and anomaly segments. Fig. 2 shows the framework of the proposed method, i.e. the OPE-GSM. In the pre-processing step, the raw time series is split into subsequences with a sliding window. Each window of data is regarded as an independent sample and fed into the OPE-GSM. The OPE-GSM method contains three parts: (1) The BiLSTM-based one-step predictive encoder, which encodes the time series from both directions and outputs the restructured errors and latent feature vectors. (2) A scoring function, which fits the errors under a Gaussian distribution and computes the negative log likelihood as an anomaly score. (3) The Gaussian segment model, which divides the latent features into segments by maximizing a cumulative gaussian likelihood function. The following subsections will detail each component of the OPE-GSM.

C. Bidirectional LSTM Encoder for long time series

The basic idea of encoder-based AD methods is to use restructured errors to determine whether the input observation is anomalous or not. Following this general idea, we propose the One-step Predictive Encoder (OPE), which predicts the subsequent observation $\boldsymbol{x}_{t+1}$ at time t rather than restructuring the present observation $\boldsymbol{x}_t$. The OPE takes advantage of BiLSTM [9, 16] to encode a time series in both positive and negative time directions. Fig. 3 shows the unfolded structure of OPE over n steps, where the target encoding sequence is $\boldsymbol{X}_e = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n]$, the forward decoding sequence is $\boldsymbol{X}_{fd} = [\boldsymbol{x}'_1, \boldsymbol{x}'_2, \cdots, \boldsymbol{x}'_n]$ and the backward decoding sequence is $\boldsymbol{X}_{bd} = [\boldsymbol{x}''_1, \boldsymbol{x}''_2, \cdots, \boldsymbol{x}''_n]$. Here, we aim to compute the restructured error for every observation in $\boldsymbol{X}_e$ and further detect if there is anomalous observation in $\boldsymbol{X}_e$ and which one is. To achieve this, we first extend the encoding sequence with a head initializer $\boldsymbol{x}_0$ and a tail initializer $\boldsymbol{x}_{n+1}$, so that we can give a prediction for $\boldsymbol{x}'_1$ and $\boldsymbol{x}''_n$, respectively. Then we feed $[\boldsymbol{x}_0, \boldsymbol{X_e}]$ into BiLSTM step by step along the positive time direction and BiLSTM outputs the forward decoding sequence $\boldsymbol{X}_{fd}$. Similarly, $[\boldsymbol{X_e}, \boldsymbol{x}_{n+1}]$ are fed into BiLSTM along the negative direction and we can get the backward decoding sequence $\boldsymbol{X}_{bd}$. Finally, OPE outputs the latent feature vector $\boldsymbol{h}$ and restructured error vector $\boldsymbol{e}$ for further anomaly detection (more details in section III-D and III-E). The $\boldsymbol{h}$ with the shape of $\mathbb{R}^{2d_h \times d_h}$ concatenates the forward hidden state $\boldsymbol{h}_f \in \mathbb{R}^{d_h \times n}$ and backward hidden state $\boldsymbol{h}_b \in \mathbb{R}^{d_h \times n}$. The $\boldsymbol{e} \in \mathbb{R}^n$ is the mean of forward and backward restructured errors, where the i-th element of $\boldsymbol{e}$ can be calculated by

$$e_i = \frac{1}{2}(||\boldsymbol{x}_i - \boldsymbol{x}'_i||_2 + ||\boldsymbol{x}_i - \boldsymbol{x}''_i||_2) \tag{2}$$

D. Anomaly points prediction

Different from references [7, 8] that using the restructured errors as anomaly scores directly, we model the errors $\{e_i\}_{i=1}^T$ with a normal distribution $\mathcal{N}(\mu, \sigma^2)$ and regard the negative log likelihood as the anomaly score. Specifically, we first estimate μ and σ using Maximum Likelihood Estimation, then calculate the negative log likelihood by $-\log(f(e_i, \mu, \sigma))$, where $f(e_i, \mu, \sigma)$ is the probability density function. Ignoring the constant, the anomaly score is

$$s_i = \frac{(e_i - \mu)^2}{2\sigma^2} \tag{3}$$

Having the anomaly scores, we simply use the 3σ principles to binary the scores where the observation locating 3σ away from the mean error will be regarded as an extreme anomaly.

E. Anomaly segment and change points prediction

The anomaly segment means a sequence of consecutive anomaly points. Since the anomaly score is calculated for each observation, it can only indicate whether a certain observation differs from the current trends or not, but fails to detect the duration of the anomaly. Furthermore, anomaly-score-based detection can not distinguish the single extreme anomaly point and the change point. Motivated by this, we apply a Guassian segment model (GSM) to detect the anomaly segments and change points.

Based on the autoencoder shown in Fig. 3, we can calculate the latent feature vector for each observation of a time series, then we predict the anomaly segment and change points. Why using the latent features rather than the raw time series? Because the time series may have seasonal periods and trends, where the observations are not independent. However, it is more reasonable to assume the encoded features follows Gaussian distribution.

We denote the latent features by $\{\boldsymbol{h}_t\}_{t=1}^T$ and assume that $\boldsymbol{h}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$, where the $\boldsymbol{\mu}_t$'s and $\boldsymbol{\Sigma}_t$'s only vary at K change points $[b_1, b_2 \cdots, b_K]$. In another word, these change points split the time series into $K+1$ segments where two consecutive segments draw from different Gaussian distributions. Here we aim to find the change points $[b_1, b_2 \cdots, b_K]$ and further detect whether each segment is anomalous or not.

The regularized log-likelihood of hidden feature sequences under SGM is

$$\ell(\boldsymbol{b}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{K+1} \sum_{t=b_{i-1}}^{b_i - 1} f(\boldsymbol{h}_t, \boldsymbol{\mu}^{(i)}, \boldsymbol{\Sigma}^{(i)}) - \lambda \mathbf{Tr}(\boldsymbol{\Sigma}^{(i)})^{-1} \tag{4}$$

where $\boldsymbol{b}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ are the lists of change points $\boldsymbol{b} = \{b_i\}_{i=1}^K$, segment means $\boldsymbol{\mu} = \{\mu_i\}_{i=1}^K$and segment covariances $\boldsymbol{\Sigma} = \{\Sigma_i\}_{i=1}^K$, respectively. $f(\boldsymbol{h}_t, \boldsymbol{\mu}^{(i)}, \boldsymbol{\Sigma}^{(i)})$ is the probability density value of $\boldsymbol{h}_t$ under $\mathcal{N}(\boldsymbol{\mu}^{(i)}, \boldsymbol{\Sigma}^{(i)})$, and the term $-\lambda \mathbf{Tr}(\Sigma^{(i)})^{-1}$ is the regulation item. If the change points are fixed, the segment means and covariances can be empirically estimated by

$$\hat{\boldsymbol{\mu}}^{(i)} = \frac{1}{b_i - b_{i-1}} \sum_{t=b_{i-1}}^{b_i - 1} \boldsymbol{h}_t \tag{5}$$

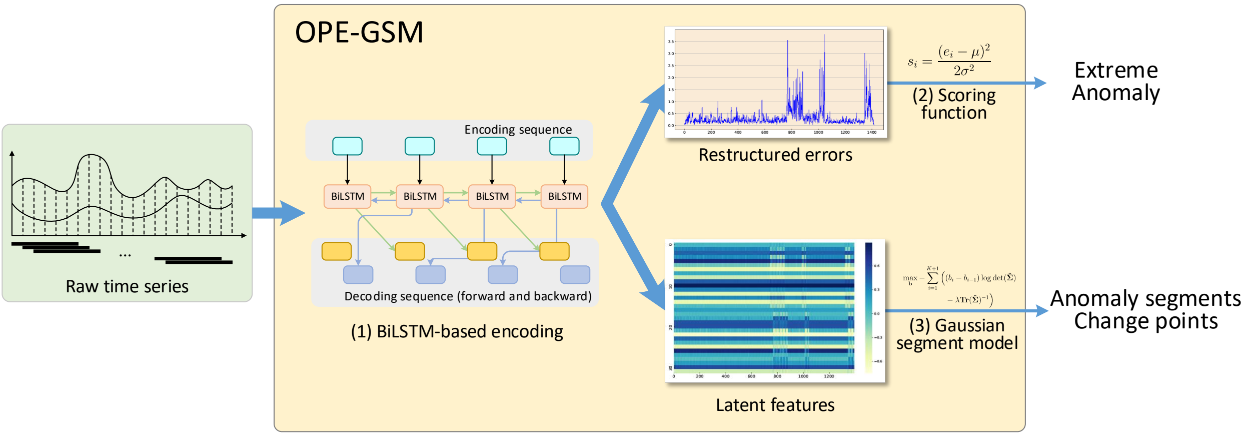

OPE-GSM
Raw time series
...
Encoding sequence
BiLSTM
Decoding sequence (forward and backward)
(1) BiLSTM-based encoding
Restructured errors
Latent features
$s_i = \frac{(e_i-\mu)^2}{2\sigma^2}$
(2) Scoring function
Extreme Anomaly
$\max_{\mathbf{b}} - \sum_{i=1}^{K+1}\Big((b_i - b_{i-1})\log\det(\hat{\mathbf{\Sigma}}) - \lambda \mathbf{Tr}(\hat{\mathbf{\Sigma}})^{-1}\Big)$
(3) Gaussian segment model
Anomaly segments
Change points

Fig. 2. The framework of OPE-GSM. The input of OPE-GSM is subsequence of observations generated by a sliding window. The outputs are the anomaly scores for extreme anomalies; the time steps for change points and anomaly segments.

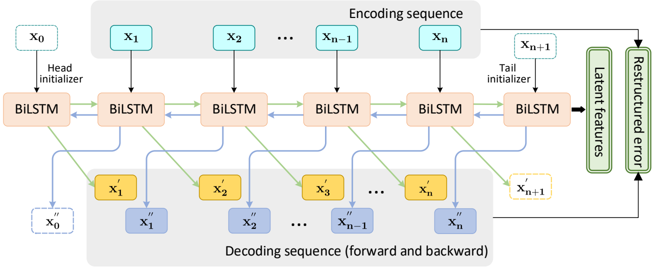


Fig. 3. BiLSTM-based one-step predictive encoder. The input is extended encoding sequence $[\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{n+1}]$. The final outputs contain latent feature vectors and the restructured error vectors. The forward and backward encodings are represented in different colored lines. The dashed boxes are not taken into account when computing the restructured errors.

$$\hat{\mathbf{\Sigma}}^{(i)} = \frac{1}{b_i - b_{i-1}} \sum_{t=b_{i-1}}^{b_i - 1} (\boldsymbol{h}_t - \boldsymbol{\mu}^{(i)})^2 + \frac{\lambda}{b_i - b_{i-1}} \boldsymbol{I} \tag{6}$$

Therefore, the log-likelihood estimation is only related to the change points $\boldsymbol{b}$ and the regularized maximum likelihood estimation problem can be formulated by

$$\max_{\boldsymbol{b}} -\frac{1}{2} \sum_{i=1}^{K+1} \Big((b_i - b_{i-1}) \log\det(\hat{\mathbf{\Sigma}}) - \lambda \mathbf{Tr}(\hat{\mathbf{\Sigma}})^{-1} \Big) \tag{7}$$

Solving the optimization problem 7 by Dynamic programming [15] or the fast greedy search algorithm in [17], we can find the time series change points. Paper [17] also gives a method to determine the number of change points K. Furthermore, we can set a threshold for segment means $\{\hat{\boldsymbol{\mu}}^{(i)}\}_{i=1}^{K}$ and segment covariances $\{\hat{\mathbf{\Sigma}}^{(i)}\}_{i=1}^{K}$ to detect the anomaly segments. The anomaly segment detection is usually application-dependent because a time series may have multi-underlying states (as shown in the Fig.7 and Fig. 8) will be discussed in the future journal version.

IV. Experimental Results

In this section, we demonstrate the performance of the proposed algorithm on two real-world TSAD dataset. The dataset information is listed in Table I. **Yahoo**[1] contains the real traffic data of Yahoo services where the anomaly points are labeled by editors manually. Following [18], we select parts of data in our experiments. **KPI**[2] consists of multi KPI curves collected from various Internet companies, such as eBay, Tencent, etc.

TABLE I
Dataset Information

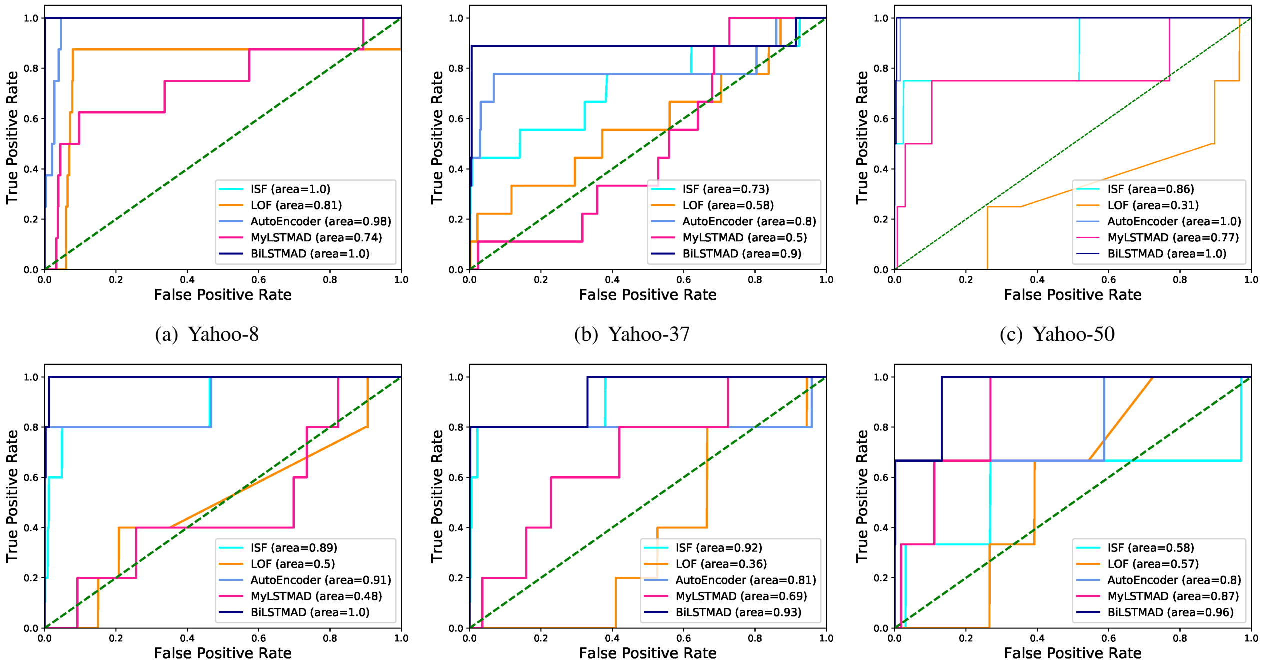
Dataset	Length	#Sequences	#Dimension	Interval	Percent(%)
Yahoo	1432.13	15	$\mathbb{R}^1$	1 hour	2.52
KPI	17568	3	$\mathbb{R}^1$	5 min	1.13

Evaluation metrics Similar to papers [19, 20], we use area-under-the-curve (AUC) and F1-score as the evaluation metrics. AUC, i.e. the area of under the receiver operating characteristic curve (ROC), evaluates the performance of the anomaly scores with a varying discrimination threshold. F1-score, being the harmonic mean of precision and recall, evaluates the performance of the anomaly decision function. Both AUC and F1-score are effective evaluation metrics for TSAD tasks because they are robust to the extremely unbalanced distribution.

Comparison methods We compare our proposed method with six competing methods, including

- Isolation Forest (ISF) [5], which isolates the outlier by a randomized clustering forest.
- Local Outlier Factor (LOF) [6], which uses the distribution density to detect anomaly points.
- One-class SVM [21], which learns a kernel-based decision frontier to describe the major samples.

[1] https://github.com/OctoberChang/klcpd_code/tree/master/data/yahoo
[2] https://github.com/shubhomoydas/ad_examples



(a) Yahoo-8

(b) Yahoo-37

(c) Yahoo-50

(d) Yahoo-54

(e) Yahoo-55

(f) Yahoo-56

Fig. 4. ROC curves of all methods on 6 randomly selected time series from **Yahoo** dataset

- Matrix Profile (MP) [3], which is the state-of-the-art similarity searching TSAD method.
- Deep autoencoder (DeepAD) [7], which uses a deep autoencoder to reconstruct subsequences of a time series.
- LSTM autoencoder (LSTMAD) [19], which uses LSTM-based autoencoder to detect the anomaly points.

Implementation Details All the algorithms are implemented in `Python 3.5`. ISF, LOF and One-class SVM are implemented using `Scikit-learn`. DeepAD and LSTMAD are reimplemented based on `https://github.com/KDD-OpenSource/DeepADoTS`. For traditional methods, we use the default parameters. For all deep learning methods, we use the Adadelta optimizer and set the learning rates to 10^{-3}. The hidden dimension of our BiLSTM is set as 16, the Regularization parameter λ in GSM is set as 1.

We report AUC and F1-score in Table II and Table III, respectively. Firstly, as can be seen from Table II, our proposed method outperforms other methods on both datasets when using the AUC metric, and achieves 3.67% better on average than the best baseline method, i.e. DeepAD. Our method's performance is also significantly better than that of LSTMAD, indicating that encoding the time series in a predictive form with BiLSTM is better than simply reconstructing with a regular LSTM. Secondly, the deep learning methods perform better than non-deep learning methods when using F1-score. The bad performance of non-deep methods may be caused by that the build-in anomaly decision function in *Scikit-learn* package is not suitable for time series data. However, the traditional methods also give comparable results on the AUC metric.

TABLE II
AUC ON TWO REAL-WORLD DATASETS

Methods	**Yahoo**	**KPI**	**Average**
ISF	0.8636	0.7679	0.8158
LOF	0.5266	0.5129	0.5197
MP	0.5707	0.4637	0.5172
SVM	0.8463	0.7254	0.6849
DeepAD	0.8759	0.7832	0.8295
LSTMAD	0.8510	0.6477	0.7500
OPE-GSM (Proposed)	**0.9072**	**0.8252**	**0.8662**

TABLE III
F1-SCORE ON TWO REAL-WORLD DATASETS

Methods	**Yahoo**	**KPI**	**Average**
ISF	0.1359	0.0333	0.0846
LOF	0.0354	0.0174	0.0264
MP	0.0717	0.0625	0.0671
SVM	0.0751	0.0311	0.0434
DeepAD	**0.3185**	0.3077	0.3131
LSTMAD	0.1436	0.0386	0.1922
OPE-GSM (Proposed)	0.3022	**0.3436**	**0.3229**

To present the results more intuitively, we randomly select 6 time-series examples from the **Yahoo** dataset and plot the ROC curves of all methods in Fig. 4. In Fig. 4, the horizontal axis is the false positive rate and the vertical axis is the true

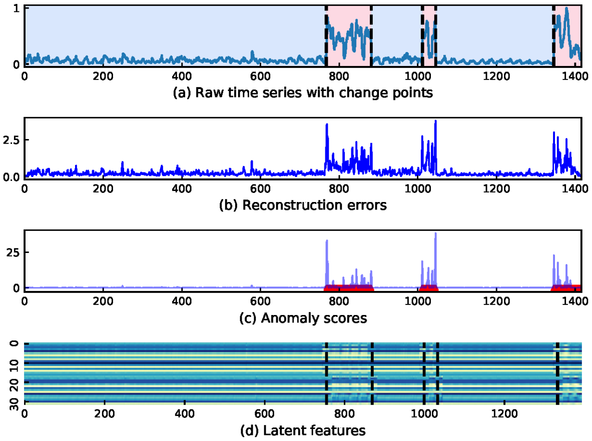


Fig. 5. Visual analysis on **Yahoo-16**

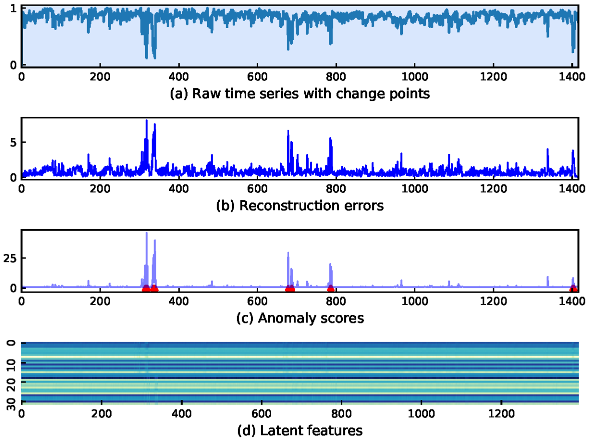


Fig. 6. Visual analysis on **Yahoo-22**

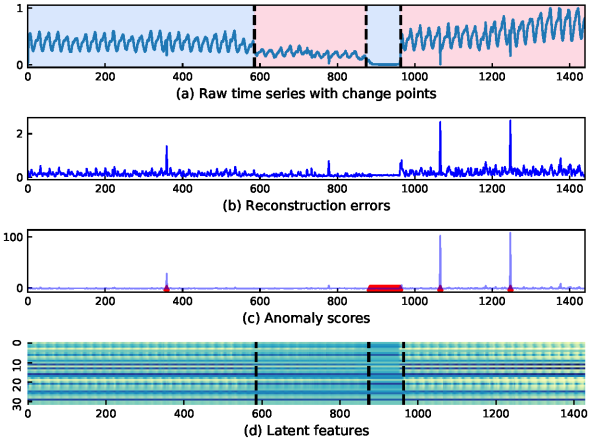


Fig. 7. Visual analysis on **Yahoo-27**

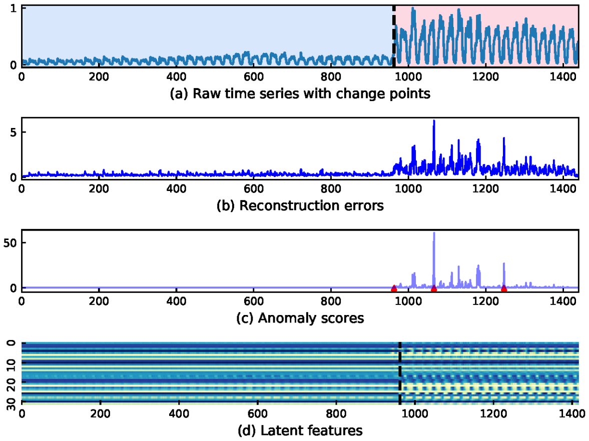


Fig. 8. Visual analysis on **Yahoo-56**

positive rate, so the area under the curve is the AUC metric. Fig. 4 shows that our method achieves the best performance in most cases, but the second-best method varies among ISF, DeepAD and LSTMAD.

We select four time series from **Yahoo** to further show how does the proposed method find the anomaly points. The selected time series include **Yahoo-16**, **Yahoo-22**, **Yahoo-27**, and **Yahoo-56**, as shown in Fig. 5, 6, 7 and 8, respectively. In all 4 figures, the subfigure (a) shows the raw time series where the vertical dashed lines indicate the change points predicted by the GSM. The subfigure (b) is the error curve computed by equation (2). The subfigure (c) presents the anomaly scores calculated by equation (3) and the red triangles label the ground-truth anomaly points. The subfigure (d) shows the 32-dimensional latent feature vectors where the vertical dashed lines are also the predicted change points.

As shown in Fig. 5-(a), **Yahoo-16** mainly has anomaly segments, where the normal segments have seasonal periods with small variance, but the anomaly segments are unpredictable with large variance. The subfigure (b) and (c) show that most observations in the anomaly segments have larger restructured errors and anomaly scores than those in the normal segments, but parts of the anomalous observations still have low scores. Therefore, only score-based detection can not find anomaly segments. However, as can be seen in subfigure (d), the hidden features are distinguished between normal and anomalous segments and our GSM model can find the change points accurately, so simple extra criterion can help to find the anomaly segments.

Fig. 6 shows the visual analysis on the **Yahoo-22**. The **Yahoo-22** has only one underlying state and several extreme anomaly points. Although the seasonal period is not clear as **Yahoo-16**, subfigure (c) shows that the OPE model can give accurate predictions for the anomaly points. The subfigure (d) shows the latent features keeps consistent throughout the time series and no change points are found.

Fig. 7 shows the visual analysis on **Yahoo-27**. Different from other examples, **Yahoo-27** have a segment that always equals to 0. This is caused by the missing data, i.e. the monitoring system does not receive any data during this period. From

Fig. 7-(b) and (c), we can see that this 'null' segment's errors and anomaly scores also nearly equal to zero, indicating that the encoder-based method can not detect it. Actually, all other methods can neither find it directly. However, our Gaussian segment model can detect the beginning and end of this 'null' segment, so we can further detect this anomaly segment with a simple extra criterion.

As shown in Fig. 8-(a), **Yahoo-56** has two normal states, one with small fluctuations that starts from beginning to around 960 timesteps and the other with relatively large fluctuations that starts from around 960 to the end. Therefore, one change point happens at the junction of two states. From Fig. 8-(a), we can see that our segment model can find the change point, although, the dataset maker hasn't labeled it as an anomaly point. Fig. 8-(d) also shows that the bidirectional OPE encodes the two states into different hidden features.

TABLE IV
AUC BY UNIDIRECTIONAL AND BIDIRECTIONAL OPE

Datasets	**Forward**	**Backward**	**Bidirectional**	**Improvement**
Yahoo	0.8281	0.8828	**0.8928**	4.35%
KPI	0.6923	0.6402	**0.7854**	17.88%

We also compare the performances of bidirectional OPE and unidirectional OPE to demonstrate that bidirectional OPE is better for the TSAD task. Table IV shows the results where the **Forward** means using a unidirectional LSTM to encode the time series in the positive time direction, and the **Backward** means encoding in the negative direction. For a fair comparison, we double the number of parameters for the unidirectional-LSTM, so that the three models in Table IV have a similar amount of parameters. As can be seen from Table IV, bidirectional OPE outperforms the unidirectional OPE by 4.35%, 17.88%, respectively. The reason for the advantage of bidirectional encoding over unidirectional encoding is that the structure of BiLSTM enables it to use all available information in the past and future of a specific time window. In contrast, the unidirectional LSTM is usually disturbed at the anomaly point, failing it to encode subsequent points in a time window, and regarding them as anomaly points wrongly. To sum up, BiLSTM can locate the anomaly points more accurately than regular LSTM.

V. Conclusion

In this paper, we propose a novel unsupervised time series anomaly detection method, referred to as OPE-GSM, which can find all three kinds of anomaly points defined in III-A, including extreme anomaly, change points and anomaly segment. Specially, we propose a novel BiLSTM-based one-step predictive encoder that can reconstruct the time more accurately than regular LSTM and deep autoencoder. Based on the latent features generated by OPE, we apply a SGM model to predict the change points and anomaly segments and use the negative log likelihood to detect the extreme anomaly points. Experiments on two real-world datasets demonstrate the effectiveness of the proposed method. In the future, we attempt to generate our method to online time series anomaly detection.

References

[1] J. L. P. Lima, D. Macêdo, and C. Zanchettin, "Heartbeat anomaly detection using adversarial oversampling," in *International Joint Conference on Neural Networks, IJCNN 2019 Budapest, Hungary, July 14-19, 2019*, 2019, pp. 1–7.

[2] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 3009–3017.

[3] Y. Zhu, C.-C. M. Yeh, Z. Zimmerman, K. Kamgar, and E. Keogh, "Matrix profile xi: Scrimp++: time series motif discovery at interactive speeds," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 837–846.

[4] O. Salem, A. Guerassimov, A. Mehaoua, A. Marcus, and B. Furht, "Anomaly detection in medical wireless sensor networks using svm and linear regression models," *International Journal of E-Health and Medical Communications (IJEHMC)*, vol. 5, no. 1, pp. 20–45, 2014.

[5] P. J. Rousseeuw and K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.

[6] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[7] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier detection using replicator neural networks," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 2002, pp. 170–180.

[8] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier detection for time series with recurrent autoencoder ensembles," in *28th international joint conference on artificial intelligence*, 2019.

[9] T. Ergen and S. S. Kozat, "Unsupervised anomaly detection with lstm neural networks," *IEEE transactions on neural networks and learning systems*, 2019.

[10] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[11] S. Arshi, L. Zhang, and R. Strachan, "Prediction using LSTM networks," in *International Joint Conference on Neural Networks, IJCNN 2019 Budapest, Hungary, July 14-19, 2019*, 2019, pp. 1–8.

[12] J. Zhao, F. Deng, Y. Cai, and J. Chen, "Long short-term memory - fully connected (lstm-fc) neural network for pm2.5 concentration prediction," *Chemosphere*, vol. 220, pp. 486 – 492, 2019.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 187–196.

[15] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, p. 107299, 2019.

[16] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[17] D. Hallac, P. Nystrup, and S. Boyd, "Greedy gaussian segmentation of multivariate time series," *Advances in Data Analysis and Classification*, vol. 13, no. 3, pp. 727–751, 2019.

[18] W.-C. Chang, C.-L. Li, Y. Yang, and B. Póczos, "Kernel change-point detection with auxiliary deep generative models," in *International Conference on Learning Representations*, 2019.

[19] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[20] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," 2018.

[21] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.