4th Asia Pacific Conference of the Prognostics and Health Management,
Tokyo, Japan, September 11 – 14, 2023 R03-04

Unsupervised Retrieval Based Multivariate Time Series Anomaly Detection and Diagnosis with Deep Binary Coding Models

Takehiko Mizoguchi, Yuji Kobayashi, and Yasuhiro Ajiro

NEC Corporation, Kawasaki, Kanagawa, Japan
{*tmizoguchi, y_koba, y.ajiro*}*@nec.com*

ABSTRACT

Retrieval based multivariate time series anomaly detection and diagnosis refer to identifying abnormal status in certain time steps and pinpointing the root cause input variables, i.e., sensors, by comparing a current time series segment and its relevant ones that are retrieved from huge amount of historical data. Binary coding with a deep neural network can be applied to reduce the computational cost of the retrieval tasks. However, it is hard to pinpoint the root cause sensors that are responsible for the anomaly, once multivariate time series segments are transformed into binary codes. In this paper, we present an unsupervised retrieval based multivariate time series anomaly detection and diagnosis method with deep binary coding model, to secure both efficiency and explainability. Specifically, we first transform input multivariate time series segments into low dimensional features with a temporal encoder. Subsequently, two hash functions predict two binary codes with different lengths from each feature. The binary codes with two different lengths can contribute to accelerate both anomaly detection and anomaly diagnosis. Experiments performed on datasets from various domains including real optical network, demonstrate the effectiveness and efficiency of the proposed method.

1. INTRODUCTION

Multivariate time series data naturally arises in many areas of real world applications. For example, complex physical systems such as power plants, optical networks are equipped with a large number of sensors distributed across different components to monitor the operation status in real-time. Moreover, due to the recent widespread of wearable devices, such sensors, which simultaneously record various status at regular intervals, could be placed even on human bodies for continuously monitoring our health status (Sprint, Cook, Weeks, Dahmen, & Fleur, 2017). Intuitively, huge amount of historical multivariate time series recorded from a system can be useful to detect unusual anomaly status. For example, when the system shows some faults, multivariate time series should be dissimilar from any of historical cases. Therefore, anomaly detection based on multivariate time series retrieval, i.e., retrieving multivariate time series segments (a slice of multivariate time series that lasts for a short time period) from database by querying with a current segment –it is called *retrieval based multivariate time series anomaly detection*– is an important problem.

A naïve approach to multivariate time series retrieval is to measure the pair-wise similarity of multivariate time series in the raw input space based on, e.g., Euclidean distance or Dynamic Time Warping (DTW) (Rakthanmanon et al., 2012). This could be the most accurate in terms of the comparison between raw time series segments, but unfortunately, it is usually computationally infeasible if the number or the length of time series are large. A promising approach to this problem is to obtain a good representation of time series segments (Chakrabarti, Keogh, Mehrotra, & Pazzani, 2002) while most of existing technique consider only univariate time series and require domain knowledge about target systems. In recent years, many methods based on approximate nearest neighbor (ANN) search with deep neural networks, e.g., Convolutional Neural Network (CNN) based method (Yang, Lin, & Chen, 2018) and Recurrent Neural Network (RNN) based method (Song, Xia, Cheng, Chen, & Tao, 2018; Zhu et al., 2020), have been emerged as the leading approaches.

However, even if effective binary codes are available by such state-of-the-art hashing methods, to practically retrieve historical multivariate time series from a query, we still need two heavy load processes; calculating pair-wise similarity between all pairs of a query and huge amount of historical data, and sorting based on the similarity. Moreover, once multivariate time series segments are transformed into binary codes, it is hard to pinpoint the root cause input variable that is responsible for the anomaly.

To address aforementioned issues, in this paper, we present Deep Hashing Network for Retrieval based Anomaly Detection (DHN-RAD) to perform unsupervised multivariate time series anomaly detection serving both efficiency and explainability. DHN-RAD employs the Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) units to extract low dimensional features from the input time series segments capturing their temporal dynamics. Two hash functions predict two different length binary codes from each feature. Triplet losses for these two binary codes are employed in unsupervised way to simultaneously preserve relative similarity relations only with the nearest neighbor information in the original input space. In query phase, we perform sub-linear search that requires searching only small subset of historical data just by comparing shorter sub-linear binary codes. We also propose anomaly detection and diagnosis methods fully utilizing binary codes. Anomaly detection is performed based on the retrieval result, and once anomaly is detected at a certain time point, sensor ranking for anomaly diagnosis can be done efficiently by comparing query time series segments and small number of exemplar segments selected with sub-linear binary codes.

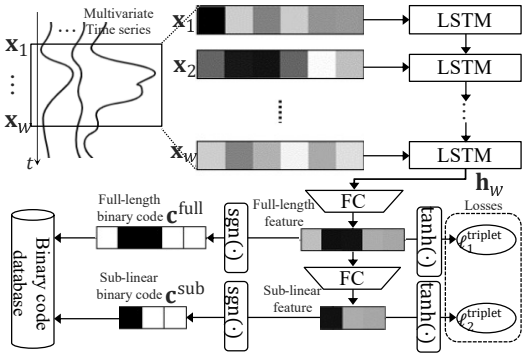

Figure 1. The network architecture of proposed DHN-RAD

Experiments are performed in the context of multivariate time series anomaly detection in various domains including IoT system state monitoring, human activity monitoring and optical network monitoring. They demonstrate the effectiveness and efficiency of the proposed method.

2. Unsupervised Deep Hashing Network for Retrieval Based Anomaly Detection

In this section, we first state the problem of unsupervised multivariate time series retrieval. Then, we present a layer-by-layer description of our proposed Deep Hashing Network for Retrieval based Anomaly Detection (DHN-RAD) model with the strategy for efficient time series retrieval based on two different length binary codes. Figure 1 illustrates the overall architecture.

2.1. Problem Statement

We introduce some notations used in this paper. We denote a multivariate time series segment $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^d]^\top = [\mathbf{x}_1, \ldots, \mathbf{x}_w] \in \mathbb{R}^{d \times w}$ as a d-dimensional and w-length segment, where w is the length of window, $\mathbf{x}^\ell = [x_1^\ell, x_2^\ell, \ldots, x_w^\ell] \in \mathbb{R}^w$ $(\ell = 1, 2, \ldots, d)$ is a time series segment of length w for the ℓ-th dimension (sensor), $\mathbf{x}_t = [x_t^1, x_t^2, \ldots, x_t^d] \in \mathbb{R}^d$ $(t = 1, 2, \ldots, w)$ is a vector from all d dimensions of time series segment at a certain time point t.

Suppose that we have a collection of historical time series segments denoted by $\mathcal{D} = \{\mathbf{X}_i\}_{i=1}^N$, where N is the total number of segments in the collection. Given a newly incoming multivariate time series segment query $\mathbf{X}_q \notin \mathcal{D}$, i.e., a slice of d-dimensional time series which lasts w time steps, the time series retrieval task is to find its most similar time series segments in $\mathcal{D}$, i.e., we aim to obtain

$$\mathbf{X}_q^\star \in \underset{\mathbf{X}_p \in \mathcal{D}}{\arg\max}\, \mathcal{S}(\mathbf{X}_q, \mathbf{X}_p), \tag{1}$$

where p is the index of p-th segment $(p \in \{1, 2, \ldots, N\})$ and $\mathcal{S} : \mathbb{R}^{d \times w} \times \mathbb{R}^{d \times w} \to \mathbb{R}$ is a function which measures the similarity between two multivariate time series segments.

2.2. Feature Extraction

To perform multivariate time series retrieval efficiently, it is essential problem to obtain a good simple representation of raw multivariate time series segments capturing their temporal dynamics. Given a multivariate time series segment $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d] \in \mathbb{R}^{d \times w}$, where $\mathbf{x}_t \in \mathbb{R}^d$ $(1 \le t \le d)$, we aim to learn a non-linear feature extraction function $F : \mathbb{R}^{d \times w} \to \mathbb{R}^m$ from $\mathbf{X}$ to m-dimensional $(m \ll d \times w)$ representation (feature) $\mathbf{h} \in \mathbb{R}^m$ with

$$\mathbf{h} := F(\mathbf{X}). \tag{2}$$

To extract features from multivariate time series segments, we have several choices such as Convolutional Neural Network (CNN) (N. Kalchbrenner, 2014), Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) and Transformer (Vaswani et al., 2017). In this paper, we employ LSTM as an example as F, since it explicitly captures both the temporal dynamics and the long-term dependencies of inputs, and have been widely used for sequence to sequence learning in many areas (Cho et al., 2014). In the feature extraction, the last hidden state of LSTM units is employed as the feature of a raw multivariate time series segment, since it encodes temporal dynamic information in the entire segment. Finally, F can be written as:

$$F(\mathbf{X}) = \mathrm{LSTM}(\mathbf{X}; \theta_{\mathrm{LSTM}}), \tag{3}$$

where θ_{LSTM} is the set of trainable parameters in LSTM.

2.3. Feature-Binary Layer

In feature-binary layer, we aim to extract two kinds of binary codes with different length, v_1-bits full-length binary codes and v_2-bits sub-linear binary codes $(v_1 > v_2)$ from the output of feature extraction layer.

2.3.1. Binary Code Prediction Functions

Given the representation for a raw multivariate time series segment $\mathbf{h}$, we aim to learn two mappings $H_1 : \mathbb{R}^m \to \{-1, +1\}^{v_1}$ and $H_2 : \mathbb{R}^m \to \{-1, +1\}^{v_2}$ which compress m-dimensional real-valued input $\mathbf{h}$ into respectively v_1-bit and v_2-bit binary codes. These mappings are known as whole *binary embedding* or *hash functions* in the literature and are expressed as

$$H_i(\mathbf{h}) = \mathrm{sgn}(G_i(\mathbf{h})), \quad (i = 1, 2), \tag{4}$$

where $\mathrm{sgn}(\cdot)$ is the element-wise sign function that extracts the sign of each element in the input, and $G_i : \mathbb{R}^m \to \mathbb{R}^{v_i}$ $(i = 1, 2)$ is a prediction function. A variety of prediction function are available for serving to specific data domains and practical applications. In this paper, for simplicity, we utilize linear prediction functions for G_1 and G_2, i.e.,

$$G_1(\mathbf{h}; \mathbf{W}_1) := \mathbf{W}_1(\mathbf{h} - \bar{\mathbf{h}}), \tag{5}$$

$$G_2(\mathbf{h}; \mathbf{W}_2) := \mathbf{W}_2(G_1(\mathbf{h}) - \bar{\mathbf{g}}), \tag{6}$$

where $\mathbf{W}_1 \in \mathbb{R}^{v_1 \times m}$, $\mathbf{W}_2 \in \mathbb{R}^{v_2 \times v_1}$ are weight matrices to be learned, and biases $\bar{\mathbf{h}} := \frac{1}{N}\sum_{i=1}^N F(\mathbf{X}_i)$, $\bar{\mathbf{g}} := \frac{1}{N}\sum_{i=1}^N G_1(F(\mathbf{X}_i))$ are set for making each bit nearly balanced, to take as much information as possible (Gong, Lazebnik, Gordo, & Perronnin, 2013). From Eqs. (4), (5) and (6), we can summarize whole hash functions H_1 and H_2 as:

$$H_i(\mathbf{h}; \mathbf{W}_i) := \mathrm{sgn}(G_i(\mathbf{h}; \mathbf{W}_i)), \quad (i = 1, 2),$$

which are parameterized respectively by $\mathbf{W}_1$ and $\mathbf{W}_2$. In the following description, we simply use $H_1(\mathbf{h})$ and $H_2(\mathbf{h})$ for denoting $H_1(\mathbf{h}; \mathbf{W}_1)$ and $H_2(\mathbf{h}; \mathbf{W}_2)$, respectively.

2.3.2. Unsupervised Triplet Losses

Desired hash functions should keep relative similarity relationship in output Hamming space between two binary codes from that between two multivariate time series in input space. Motivated by (Schroff, Kalenichenko, & Philbin, 2015), we leverage relative similarities in the form of triplets $(a, +, -) \in \mathcal{I}_{\text{triplet}}$, where a, $+$ and $-$ stand for *anchor*, *positive* and *negative* indices, respectively, and $\mathcal{I}_{\text{triplet}}$ is the set of all possible triplet indices. In this paper, the triplets are selected based on the similarity in the original input space, e.g., $(a, +, -)$ are selected so that $\mathbf{X}_+$ is within k-nearest neighbor (k-NN) from $\mathbf{X}_a$ while $\mathbf{X}_-$ is out of k-NN from $\mathbf{X}_a$, respectively in the input space.

Intuitively, the desired hash functions $H_i(\cdot)$ $(i = 1, 2)$ would be expected to preserve these relative similarity relationships revealed by $\mathcal{I}_{\text{triplet}}$ within the Hamming space, i.e., to make Hamming distance between the embeddings $H_i(\mathbf{h}_a)$ and $H_i(\mathbf{h}_+)$ smaller than that between $H_i(\mathbf{h}_a)$ and $H_i(\mathbf{h}_-)$, where $\mathbf{h}_a$, $\mathbf{h}_+$ and $\mathbf{h}_-$ are respectively anchor, positive and negative features extracted from $\mathbf{X}_a$, $\mathbf{X}_+$ and $\mathbf{X}_-$ by $F(\cdot)$ in Eq. (2). The triplet losses that evaluate hash functions H_i $(i = 1, 2)$ under above intuition are then

$$\ell_i^{\text{triplet}} := \sum_{(a,+,-)\in\mathcal{I}_{\text{triplet}}} \max(0, d_i^+ - d_i^- + \alpha), \qquad (7)$$

where $d_i^q := \|H_i(\mathbf{h}_a) - H_i(\mathbf{h}_q)\|_0$ is the Hamming distance between $H_i(\mathbf{h}_a)$ and $H_i(\mathbf{h}_q)$ $(q \in \{+, -\})$, $\|\mathbf{h}\|_0$ is the ℓ_0-norm, which counts the number of non-zero entries in $\mathbf{h}$, and $\alpha \geq 0$ is a margin.

2.4. Optimization Problem

Based on the discussion in Sections 2.3, we can summarize the loss function as the following objective of our proposed network model:

$$\ell(\theta) := \ell_1^{\text{triplet}}(\theta) + \lambda \ell_2^{\text{triplet}}(\theta), \qquad (8)$$

where θ is the set of all trainable parameters in the model, i.e., $\theta := \theta_{\text{LSTM}} \cup \{\mathbf{W}_1, \mathbf{W}_2\}$, and $\lambda \geq 0$ is the weight parameter that controls the importance of the triplet loss ℓ_2^{triplet} for sub-linear binary codes.

Unfortunately, our objective (8) is hard to be optimized as it is, since the hash functions $H_i(\cdot)$ $(i = 1, 2)$ are discrete mappings and the Hamming distances in the triplet loss ℓ_i^{triplet} lies in a discrete space. To address this issues, we relax the original discrete objective to a continuous and differentiable surrogate. The hash functions $H_i(\cdot)$ $(i = 1, 2)$ can be relaxed as $H_i(\mathbf{h}) \approx \overline{H}_i(\mathbf{h}) := \tanh(G_i(\mathbf{h}; \mathbf{W}_i))$, which are differentiable, by approximating $\text{sgn}(\cdot) \approx \tanh(\cdot)$. We also relax the Hamming distance in (7) to the ℓ_1-distance, i.e., $d_i^q \approx \overline{d}_i^q := \|\overline{H}_i(\mathbf{h}_a) - \overline{H}_i(\mathbf{h}_q)\|_1$ $(q \in \{+, -\})$.

Based on the above relaxations, we finally have the following continuous and differentiable objective:

$$\overline{\ell}(\theta) := \overline{\ell}_1^{\text{triplet}}(\theta) + \lambda \overline{\ell}_2^{\text{triplet}}(\theta), \qquad (9)$$

where $\overline{\ell}_i^{\text{triplet}} := \sum_{(a,+,-)\in\mathcal{I}_{\text{triplet}}} \max(0, \overline{d}_i^+ - \overline{d}_i^- + \alpha)$ $(i = 1, 2)$. These relaxations have been naturally used for the optimization of binary embedding networks (Lai, Pan, Liu, & Yan, 2015). For optimizing the all trainable parameters θ of the proposed network, we employ Adam optimizer to perform backpropagation over entire network based on stochastic gradient descent with mini-batch size 256.

3. Retrieval Based Multivariate Time Series Anomaly Detection and Diagnosis

3.1. Time Series Retrieval with Sub-linear Search

If the training is finished, we extract two different length of binary codes $\mathbf{c}_i^{\text{full}} \in \{-1, +1\}^{v_1}$ and $\mathbf{c}_i^{\text{sub}} \in \{-1, +1\}^{v_2}$ for all historical time series segments $\mathbf{X}_i \in \mathcal{D}$ $(i = 1, \ldots, N)$ respectively by $\mathbf{c}_i^{\text{full}} := H_1(F(\mathbf{X}_i))$ and $\mathbf{c}_i^{\text{sub}} := H_2(F(\mathbf{X}_i))$. Since $v_2 < v_1$, the number of unique sub-linear binary codes $\mathbf{c}_i^{\text{sub}}$ extracted from $\mathbf{X}_i$ are expected to be much less than that of unique full-length binary codes $\mathbf{c}_i^{\text{full}}$, i.e., many different full-length binary codes would share the same sub-linear binary code. This fact enable us to perform efficient multivariate time series retrieval by sub-linear search.

The sub-linear search algorithm for efficient multivariate time series retrieval is summarized in Algorithm 1. After extracting full-length and sub-linear binary codes for all historical time series segments, we construct a sub-linear dictionary $\mathcal{I}$ which returns the set of all indices that share a common sub-linear binary code, i.e.,

$$\mathcal{I}(\mathbf{c}^{\text{sub}}) := \left\{ i \,\middle|\, \mathbf{c}_i^{\text{sub}} = \mathbf{c}^{\text{sub}} \right\} \subset \{1, \ldots, N\}. \qquad (10)$$

For a query time series segment $\mathbf{X}_q$, we extract its full-length and sub-linear binary codes, $\mathbf{c}_q^{\text{full}}$ and $\mathbf{c}_q^{\text{sub}}$ by DHN-RAD (line 2). Then, we first retrieve indices of time series segment in database by $\mathcal{I}(\mathbf{c}_q^{\text{sub}})$ and add them to the candidate indices $\mathcal{J}$ (lines 4-6 for $r = 0$). If we do not retrieve sufficient number of indices, i.e., $|\mathcal{J}| < k$, we next look for $\mathcal{I}$ with the second nearest sub-linear binary codes, i.e., Ω_r with sub-linear binary codes, $r(\geq 1)$ of whose bits are flipped from $\mathbf{c}_q^{\text{sub}}$. We iterate this process incrementing r until enough candidates are retrieved (i.e., $|\mathcal{J}| \geq k$) up to the pre-defined maximum number of flipped bits $r_{\max}$ (lines 3-7).

Once we have enough number of candidate indices, we calculate pair-wise Hamming distances Δ between full-length binary code of the query segment $\mathbf{c}_q^{\text{full}}$ and those of the subset

Algorithm 1: Top-k sub-linear search for efficient multivariate time series retrieval

Input : $\mathbf{X}_q$, $\mathcal{I}$, k, $r_{\max}$
Output: Top-k similar time series segments to $\mathbf{X}_q$

1 $\mathcal{J} \leftarrow \varnothing$, $r \leftarrow 0$;
2 $\mathbf{c}_q^{\text{full}} \leftarrow H_1(F(\mathbf{X}_q))$, $\mathbf{c}_q^{\text{sub}} \leftarrow H_2(F(\mathbf{X}_q))$;
3 **while** $|\mathcal{J}| < k$ ***and*** $r < r_{\max}$ **do**
4 $\Omega_r \leftarrow \{\mathbf{c} \in \{-1, +1\}^{v_2} | \|\mathbf{c} - \mathbf{c}_q^{\text{sub}}\|_0 = r\}$;
5 **for** $\mathbf{c}' \in \Omega_r$ **do**
6 $\mathcal{J} \leftarrow \mathcal{J} \cup \mathcal{I}(\mathbf{c}')$;
7 $r \leftarrow r + 1$;
8 $\Delta \leftarrow \{\|\mathbf{c}_j^{\text{full}} - \mathbf{c}_q^{\text{full}}\|_0 | j \in \mathcal{J}\}$;
9 $[i_1^\star, \ldots, i_k^\star] \leftarrow \text{argsort}(\Delta)[:k]$;
10 **return** $\mathbf{X}_{i_1^\star}, \ldots, \mathbf{X}_{i_k^\star}$

of database segments assigned by $\mathcal{J}$ (line 8). Then, we sort Δ in ascending order and retrieve up to k number of indices from the top ones (line 9), for example, we retrieve j' as $i_1^\star$ if $\|\mathbf{c}_{j'}^{\text{full}} - \mathbf{c}_q^{\text{full}}\|_0$ is the smallest within Δ. Finally, we retrieve k time series segments $\mathbf{X}_{i_1^\star}, \ldots, \mathbf{X}_{i_k^\star}$ as the most relevant ones.

3.2. Anomaly Detection and Diagnosis

Once we retrieve the top-k relevant time series segments $\mathbf{X}_{i_1^\star}, \ldots, \mathbf{X}_{i_k^\star}$ to a query time series segment $\mathbf{X}_q$, we can calculate the anomaly score $a(\mathbf{X}_q)$ as the aggregate of Euclidean distances of features to these top-k ones in the database. For example, if we consider the average of distances, it can be written as:

$$a(\mathbf{X}_q) := \frac{1}{k} \sum_{j=1}^{k} \|F(\mathbf{X}_{i_j^\star}) - F(\mathbf{X}_q)\|_2.$$

If an anomaly is detected in a query time series segment $\mathbf{X}_q$, i.e., $a(\mathbf{X}_q)$ exceeds a prefixed threshold $\eta > 0$, then we do sensor ranking to pinpoint which sensors (dimensions) of $\mathbf{X}_q$ are responsible for the anomaly. To measure how each sensor value is diverse between a pair of time series segments, we first define a *divergence score* of sensor ℓ ($\ell = 1, \ldots, d$) ds^ℓ between two time series segments $\mathbf{X}_p$ and $\mathbf{X}_q$:

$$ds^\ell(\mathbf{X}_p, \mathbf{X}_q) := |\bar{\mathbf{x}}_p^\ell - \bar{\mathbf{x}}_q^\ell|,$$

where $\bar{\mathbf{x}}_p^\ell$ is the average of the ℓ-th dimension of time series segment $\mathbf{X}_p$ over all w points in the window, i.e., $\bar{\mathbf{x}}_p^\ell := \frac{1}{w}\sum_{t=1}^{w} x_{t,p}^\ell$. Intuitively, 'abnormal sensors' are diverse from normal sensors of any historical time series segments, so ideally, we want to calculate the divergence score against all historical segments for each query time series segment. However, if we have huge amount of historical time series segment as assumed in this paper, such strategy is computationally infeasible. To address this issue, we propose to compare the query segment with only small subset of time series segments, i.e., *exemplars*, which summarize well whole historical data.

To select exemplars from whole historical time series segments, we use the sub-linear dictionary $\mathcal{I}$ constructed in Eq. (10), Section 3.1. Note that the number of unique sub-linear binary codes is at most 2^{v_2}. If that number is enough small comparing to the total number of segments in the database N, i.e., $2^{v_2} \ll N$, the sub-linear binary codes can be regarded as cluster assignments since many segments share a common sub-linear binary code. In this paper, we select a segment which is the closest to the centroid of a cluster assigned by a sub-linear binary code $\mathbf{c}$ as an exemplar $\mathbf{X}_\mathbf{c}^*$, i.e.,

$$\mathbf{X}_\mathbf{c}^* := \arg\min_{\mathbf{X} \in \mathcal{X}_\mathbf{c}} \|\bar{\mathbf{X}}_\mathbf{c} - \mathbf{X}\|_F,$$

where $\mathcal{X}_\mathbf{c} := \{\mathbf{X}_i | i \in \mathcal{I}(\mathbf{c})\}$ and $\bar{\mathbf{X}}_\mathbf{c} := \frac{1}{|\mathcal{X}_\mathbf{c}|}\sum_{\mathbf{X}\in\mathcal{X}_\mathbf{c}} \mathbf{X}$ and $\|\cdot\|_F$ is the Frobenius norm of matrices.

Anomaly sensors should have large divergence score for any exemplars, so we compute the sensor score s^ℓ ($\ell = 1, \ldots, d$) for a query time series segment $\mathbf{X}_q$ by

$$s^\ell(\mathbf{X}_q) := \min_{\mathbf{X}^* \in \mathcal{X}^*} ds^\ell(\mathbf{X}^*, \mathbf{X}_q),$$

where $\mathcal{X}^*$ is the set of all exemplars. Then we can get sensor ranking $\mathbf{r}$ by sorting the indices of sensors in descending order with the sensor score s^ℓ, i.e., $\mathbf{r} := [\ell_1^\star, \ldots \ell_d^\star]$, where $s^{\ell_1^\star}(\mathbf{X}_q) > \cdots > s^{\ell_d^\star}(\mathbf{X}_q)$.

Following above discussions, the proposed anomaly detection and diagnosis algorithm can be summarized in Algorithm 2.

Algorithm 2: Anomaly detection and diagnosis

Input : $\mathbf{X}_q$, $\mathcal{I}$, k, $r_{\max}$, η, $\mathcal{X}^*$
Output: Anomaly score $a(\mathbf{X}_q)$, sensor ranking $\mathbf{r}$

1 Retrieve $\mathbf{X}_{i_1^\star}, \ldots, \mathbf{X}_{i_k^\star}$ from $\mathbf{X}_q$ with $\mathcal{I}$, k and $r_{\max}$ by Algorithm 1;
2 $a(\mathbf{X}_q) \leftarrow \frac{1}{k}\sum_{j=1}^{k} \|F(\mathbf{X}_{i_j^\star}) - F(\mathbf{X}_q)\|_2$;
3 $\mathbf{r} \leftarrow \varnothing$;
4 **if** $a(\mathbf{X}_q) > \eta$ **then**
5 **for** $\ell = 1, \ldots, d$ **do**
6 **for** $\mathbf{X}^* \in \mathcal{X}^*$ **do**
7 $ds^\ell(\mathbf{X}^*, \mathbf{X}_q) \leftarrow |\bar{\mathbf{x}}^{*\ell} - \bar{\mathbf{x}}_q^\ell|$
8 $s^\ell(\mathbf{X}_q) \leftarrow \min_{\mathbf{X}^* \in \mathcal{X}^*} ds^\ell(\mathbf{X}^*, \mathbf{X}_q)$
9 $\mathbf{r} \leftarrow [\ell_1^\star, \ldots \ell_d^\star]$, where $s^{\ell_1^\star}(\mathbf{X}_q) > \cdots > s^{\ell_d^\star}(\mathbf{X}_q)$;
10 **return** $a(\mathbf{X}_q)$, $\mathbf{r}$

4. EXPERIMENTS

4.1. Datasets

We employ real multivariate time series datasets from three different sources, IoT, PAMAP2 and Optical Network as shown in Table 1.

Table 1. Details of three multivariate time series datasets

Dataset	# sensors	# time points
IoT	4	20,000
PAMAP2	52	376,416
Optical Network	96	13,054

IoT (Internet of Things) dataset is collected from a very simple IoT equipment with an acceleration sensor, a thermometer and a fan. Four time series including 3D accelerations and temperature are collected from the acceleration sensor and the label represents the anomaly status (normal, eccentric, break and stop) on the fan for each time point. We sample 19991 segments of length 10 with overlap 9 and anomaly detection models are trained on normal data and tested on the rest to detect three types of anomaly statuses.

PAMAP2 dataset[1] is for physical activity monitoring (Reiss & Stricker, 2012). It contains various different physical activities, performed by 9 subject wearing 3 internal measurement units (IMUs) and a heart rate monitor. Observations from IMUs and a heart rate monitor can dramatically change by the subjects, so in this experiment, we select one subject (Subject101) for simplicity. The dataset from Subject101 contains 52 time series with 376,416 time points and 13 different physical activities. We sample 37,741 segments of length 100 with

[1] https://archive.ics.uci.edu/ml/datasets/pamap2+physical+activity+monitoring

overlap 90. Anomaly detection model is trained on the segments only with daily physical activities such as 'lying', 'sitting', 'standing', 'ascending stairs' etc and tested to detect exercise activities such as 'running', 'cycling', 'Nordic walking' and 'rope jumping' as anomalies.

Optical network dataset is collected from a simple optical network system illustrated in Figure 2. It composed of three network nodes, which includes several devices such as optical transponders (TPND), switches (WA, XF) and amplifiers (WA, CA). We collect data from sensors in these devices and obtain 96 number of time series with 13,054 time points. We sample 13,045 segments of length 10 with overlap 9. We aim to detect two kinds of faults occurred in the network:

(a) Optical power degradation in the optical path between Node 1 and Node 2

(b) Optical power degradation by the fault of power adjustment function of WA in Node 2

Anomaly detection models are trained on time series segments without any faults and then tested on the rest to detect these two faults.

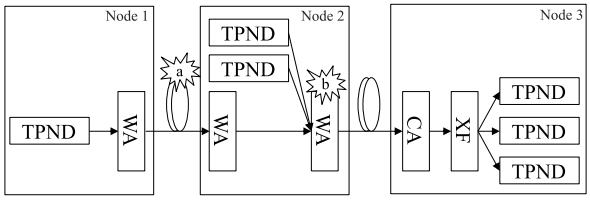

Figure 2. Network configuration diagram of optical network

4.2. Baselines and Settings

In this experiment, we compare DHN-RAD with three different representative anomaly algorithms. Among them, One-Class Support Vector Machine (OC-SVM) (Manevitz & Yousef, 2002) is general anomaly detection method and Real-Time and Self-Taught Anomaly Detection (RTST-AD) (Chen et al., 2018), which is a deep learning based method with clustering, is specialized for anomaly detection on optical network systems. Classification Score Profile (ClaSP) (Schäfer, Ermshaus, & Leser, 2021) based on self-supervision and time series segmentation, and the proposed method are specialized for time series anomaly detection. We use hash dimensions $(v_1, v_2) = (256, 16)$ and the margin $\alpha = 1.0$ for DHN-RAD. The hyper-parameter λ of DHN-RAD is optimized based on grid search over $\lambda \in \{0.001, 0.01, 0.1\}$, and threshold η for anomaly detection is determined based on the anomaly score on validation data, i.e., $\eta = \max_{\mathbf{X} \in \mathcal{X}^{\text{val}}} a(\mathbf{X}) \cdot \beta$, where $\mathcal{X}^{\text{val}}$ is the set of all validation data and $\beta > 0$ is optimized with grid search over $\beta \in \{0.8, 1.0, 1.5, 2.0\}$. DHN-RAD is implemented in Python 3.10 with PyTorch 1.9 and trained on a server with Intel® Core™ i9-7900X @ 3.3 GHz 10 core CPU and single NVIDIA GeForce RTX™ 1080 Ti graphics card.

4.3. Results

4.3.1. Anomaly Detection

The anomaly detection performances are shown in Table 2. All evaluation metrics, precision (P), recall (R) and F1 score (F1) are averaged over 5 trials with different subset of training data. It shows that the time series specialized methods ClaSP and DHN-RAD outperform general anomaly detection method because they fully utilize temporal dynamics of time series. ClaSP achieves good recall in all datasets but precision is significantly degraded in PAMAP2 and Optical Network. This is because segmenting time series gets harder as datasets become complex. DHN-RAD consistently achieves the best or the second best performance for both precision and recall and also always achieves the best F1 score for all datasets. This means the proposed method can detect as many faults as possible keeping small number of false alarm.

4.3.2. Retrieval Efficiency

We next examine the efficiency of sub-linear search (Algorithm 1) employed in our approach. Figure 3 shows the comparison of the average query time between sub-linear search and full search (i.e., calculate pair-wise hamming distance with query and all binary codes in historical database and then pick up the top k) changing $r_{\max} = 1, 2$ for each dataset. It indicates that sub-linear search is less affected by the growth of the number of examples in database than full search, resulting that the more samples in database, the more beneficial sub-linear search is.

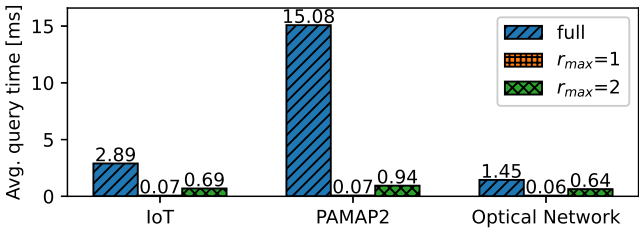

Figure 3. Retrieval efficiency comparison

4.3.3. Anomaly Diagnosis

Finally, we show the effectiveness of the proposed anomaly diagnosis algorithm (Algorithm 2) with Optical Network dataset. Figure 4 shows the time series shapes of top 5 (out of 96) sensors ranked by the proposed sensor ranking algorithm as well as anomaly score by the proposed method for both kinds of faults (a) and (b). We can clearly see that the top ranked sensors capture the change of anomaly score, and we also found that the ground truth sensors specified by domain experts are included in these top 5.

5. Conclusion

In this paper, we have proposed an unsupervised retrieval based multivariate time series anomaly detection and diagnosis method with a deep binary coding model DHN-RAD. DHN-RAD employs LSTM units to extract features from multivariate time series capturing their temporal dynamics. It extracts two kinds of binary codes of different length to perform sub-linear search for efficient multivariate time series retrieval and diagnosis. Anomaly detection and diagnosis can be done efficiently with the proposed sub-linear search, divergence score and exemplars. Experiments are performed on various datasets from different domains including IoT system monitoring, human activity monitoring and real optical network monitoring, and demonstrated the effectiveness in terms of anomaly detection, retrieval efficiency and anomaly diagnosis of the proposed method.

Table 2. Performance comparison in multivariate time series anomaly detection tasks on IoT, PAMAP2 and Optical Network datasets. The best and the second performance are indicated respectively by boldface and underline.

	IoT			PAMAP2			Optical Network		
Algorithm	P	R	F1	P	R	F1	P	R	F1
OC-SVM	0.938	0.492	0.592	0.342	0.348	0.344	0.714	0.372	0.474
RTST-AD	**1.000**	0.001	0.002	**0.738**	0.178	0.264	**0.954**	0.414	0.568
ClaSP	0.938	**1.000**	0.968	0.568	**1.000**	0.725	0.559	**1.000**	0.717
DHN-RAD (ours)	0.957	**1.000**	**0.978**	0.690	0.863	**0.760**	0.887	0.878	**0.881**

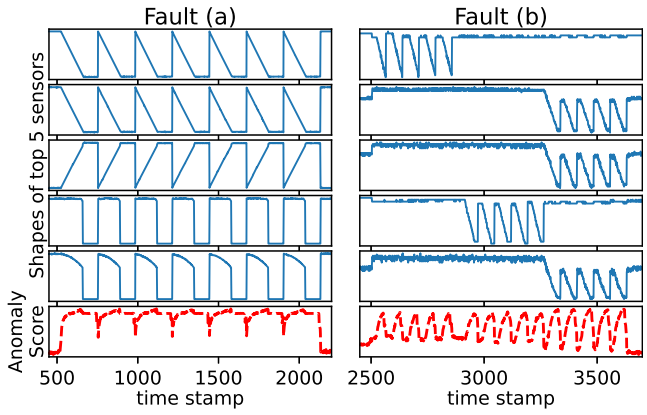

Figure 4. Time series shapes of top 5 sensors with anomaly score for the faults (a) and (b) in Optical Network dataset. The upper 5 figures for each column are the time series shape of top 5 sensors. The bottom figures with dashed lines are anomaly scores

ACKNOWLEDGEMENT

This work was obtained in part from the commissioned research (No.0470102) by National Institute of Information and Communications Technology (NICT), Japan.

REFERENCES

Chakrabarti, K., Keogh, E., Mehrotra, S., & Pazzani, M. (2002, Jan.). Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Trans. Database Systems*, *27*(2), 188–228.

Chen, X., Li, B., Shamsabardeh, M., Proietti, R., Zhu, Z., & Yoo, S. J. B. (2018). On real-time and self-taught anomaly detection in optical networks using hybrid unsupervised/supervised learning. In *Proc. Euro. Conf. Optic. Comm.*

Cho, K., Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proc. Conf. Emp. Meth. Nat. Lang. Process.*

Gong, Y., Lazebnik, S., Gordo, A., & Perronnin, F. (2013, Dec.). Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. Pat. Anal. Mach. Intel.*, *25*(12), 2916–2928.

Hochreiter, S., & Schmidhuber, J. (1997, Nov.). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Lai, H., Pan, Y., Liu, Y., & Yan, S. (2015). Simultaneous feature learning and hash coding with deep neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recog.*

Manevitz, L. M., & Yousef, M. (2002, Mar.). One-class svms for document classification. *J. Mach. Learn. Res.*, *2*, 139–154.

N. Kalchbrenner, P. B., E. Grefenstette. (2014). A convolutional neural network for modelling sentences. In *Proc. Assoc. Comp. Ling.*

Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., ... Keogh, E. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. In *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Min.*

Reiss, A., & Stricker, D. (2012). Introducing a new benchmarked dataset for activity monitoring. In *Proc. Int'l Symp. Wear. Comp.*

Schäfer, P., Ermshaus, A., & Leser, U. (2021). ClaSP - Time Series Segmentation. In *Proc. Conf. Inf. Knowl. Manage.*

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: Aunified embedding for face recognition and clustering. In *Proc. IEEE Conf. Comp. Vis. Patt. Recog.*

Song, D., Xia, N., Cheng, W., Chen, H., & Tao, D. (2018). Deep r-th root of rank supervised joint binary embedding for multivariate time series retrieval. In *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Min.*

Sprint, G., Cook, D., Weeks, D., Dahmen, J., & Fleur, A. (2017, Oct.). Analyzing sensor-based time series data to track changes in physical activity during inpatient rehabilitation. *Sensors*, *17*(10).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., ... Polosukhin, I. (2017). Attention is all you need. In *Proc. Neural Inf. Process. Syst.*

Yang, H.-F., Lin, K., & Chen, C.-S. (2018, Feb.). Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE Trans. Pat. Anal. Mach. Intel.*, *40*(2), 437–451.

Zhu, D., Song, D., Chen, Y., Lumezanu, C., Cheng, W., Zong, B., ... T. Yang, H. C. (2020). Deep unsupervised binary coding networks for multivariate time series retrieval. In *Proc. AAAI Conf. Artif. Intel.*