Unsupervised Deep Anomaly Detection for Multi-Sensor Time-Series Signals

Yuxin Zhang, Yiqiang Chen, *Senior Member, IEEE,* Jindong Wang, and Zhiwen Pan, *Member, IEEE*

Abstract—Nowadays, multi-sensor technologies are applied in many fields, e.g., Health Care (HC), Human Activity Recognition (HAR), and Industrial Control System (ICS). These sensors can generate a substantial amount of multivariate time-series data. Unsupervised anomaly detection on multi-sensor time-series data has been proven critical in machine learning researches. The key challenge is to discover generalized normal patterns by capturing spatial-temporal correlation in multi-sensor data. Beyond this challenge, the noisy data is often intertwined with the training data, which is likely to mislead the model by making it hard to distinguish between the normal, abnormal, and noisy data. Few of previous researches can jointly address these two challenges. In this paper, we propose a novel deep learning-based anomaly detection algorithm called Deep Convolutional Autoencoding Memory network (CAE-M). We first build a Deep Convolutional Autoencoder to characterize spatial dependence of multi-sensor data with a Maximum Mean Discrepancy (MMD) to better distinguish between the noisy, normal, and abnormal data. Then, we construct a Memory Network consisting of linear (Autoregressive Model) and non-linear predictions (Bidirectional LSTM with Attention) to capture temporal dependence from time-series data. Finally, CAE-M jointly optimizes these two subnetworks. We empirically compare the proposed approach with several state-of-the-art anomaly detection methods on HAR and HC datasets. Experimental results demonstrate that our proposed model outperforms these existing methods.

Index Terms—Unsupervised anomaly detection, Multi-sensor time series, Convolutional autoencoder, Attention based BiLSTM.

✦

1 INTRODUCTION

ANOMALY detection has been one of the core research areas in machine learning for decades, with wide applications such as cyber-intrusion detection [1], medical care [2], sensor networks [3], video anomaly detection [4] and so on. Anomaly detection seems to be a simple two-category classification, i.e., we can learn to classify the normal or abnormal data. However, it is also faced with the following challenges. First, training data is highly imbalanced since the anomalies are often extremely rare in a dataset compared to the normal instances. Standard classifiers try to maximize accuracy in classification, so it often falls into the trap of overlapping problem, which means that the model classifies the overlapping region as belonging to the majority class while assuming the minority class as noise. Second, there is no easy way for users to manually label each training data, especially the anomalies. In many cases, it is prohibitively hard to represent all types of anomalous behaviors. Due to above challenges, there is a growing trend to use unsupervised learning approaches for anomaly detection compared with semi-supervised and supervised learning approaches since unsupervised methods can handle the imbalanced and

unlabeled data in a more principled way [5]–[9].

Nowadays, the prevalence of sensors in machine learning and pervasive computing research areas such as Health Care (HC) [10], [11] and Human Activity Recognition (HAR) [12], [13] generate a substantial amount of multivariate time-series data. These learning algorithms based on multi-sensor time-series signals give priority to dealing with spatial-temporal correlation of multi-sensor data. Many approaches for spatial-temporal dependency amongst multiple sensors [14]–[16] have been studied. It seems intuitive to apply previous unsupervised anomaly detection methods on multi-sensor time-series data. Unfortunately, there are still several challenges.

First, anomaly detection in spatial-temporal domain becomes more complicated due to the temporal component in time-series data. Conventional anomaly detection techniques such as PCA [17], k-means [18], OCSVM [19] and Autoencoder [20] are unable to deal with multivariate time-series signals since they cannot simultaneously capture the spatial and temporal dependencies. Second, these reconstruction-based models such as Convolutional AutoEncoders (CAEs) [21] and Denoising AutoEncoders (DAEs) [22] are usually used for anomaly detection. It is generally assumed that the compression of anomalous samples is different from that on normal samples, and the reconstruction error becomes higher for these anomalous samples. In reality, being influenced by the high complexity of model and the noise of data, the reconstruction error for the abnormal input could also be fit so well by the training model [23], [24]. That is, the model is robust to noise and anomalies. Third, in order to reduce the dimensionality of multi-sensor data and detect anomalies, two-step approaches are widely adopted. As for the drawback of some works [25], [26], the joint performance of two

- *Y. Zhang is with Global Energy Interconnection Development and Cooperation Organization, Xicheng District, Beijing, China, and Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences and University of Chinese Academy of Sciences, Beijing, China. E-mail: yuxinzhang@geidco.org.*
- *Y. Chen and Z. Pan are with Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences and University of Chinese Academy of Sciences, Beijing, China. Y. Chen is also with Peng cheng Laboratory (PCL). E-mail: {yqchen, pzw}@ict.ac.cn (Corresponding author: Yiqiang Chen).*
- *J. Wang is with Microsoft Research Asia, Beijing, China, also correspondence. E-mail: Jindong.Wang@microsoft.com.*

baseline models can easily get stuck in local optima, since two models are trained separately.

In order to solve the above three challenges, this paper presents a novel unsupervised deep learning based anomaly detection approach for multi-sensor time-series data called Deep Convolutional Autoencoding Memory network (CAE-M). The CAE-M network composes of two main sub-networks: characterization network and memory network. Specifically, we employ deep convolutional autoencoder as feature extraction module, with attention-based Bidirectional LSTMs and Autoregressive model as forecasting module. By simultaneously minimizing reconstruction error and prediction error, the CAE-M model can be jointly optimized. During the training phase, the CAE-M model is trained to explicitly describe the normal pattern of multi-sensor time-series data. During the detection phase, the CAE-M model calculate the compound objective function for each captured testing data. Through combining these errors as a composite anomaly score, a fine-grained anomaly detection decision can be made. To summarize, the main contributions of this paper are four-fold:

1) The proposed composite model is designed to characterize complex spatial-temporal patterns by concurrently performing the reconstruction and prediction analysis. In reconstruction analysis, we build Deep Convolutional Autoencoder to fuse and extract low-dimensional spatial features from multi-sensor signals. In prediction analysis, we build Attention-based Bidirectional LSTM to capture complex temporal dependencies. Moreover, we incorporate Auto-regressive linear model in parallel to improve the robust and adapt for different use cases and domains.

2) To reduce the influence of noisy data, we improve Deep Convolutional Autoencoder with a Maximum Mean Discrepancy (MMD) penalty. MMD is used to encourage the distribution of the low-dimensional representation to approximate some target distribution. It aims to make the distribution of noisy data close to the distribution of normal training data, thereby reducing the risk of overfitting. Experiments demonstrate that it is effective to enhance the robustness and generalization ability of our method.

3) The CAE-M is an end-to-end learning model that two sub-networks can co-optimize by a compound objective function with weight coefficients. This single-stage approach can not only streamline the learning procedure for anomaly detection, but also avoid the model getting stuck in local minimum through joint optimization.

4) Experiments on three multi-sensor time-series datasets demonstrate that CAE-M model has superior performance over state-of-the-art techniques. In order to further verify the effect of our proposed model, fine-grained analysis, effectiveness evaluation, parameter sensitivity analysis and convergence analysis show that all the components of CAE-M together leads to the robust performance on all datasets.

The rest of the paper is organized as follows. Section 2 provides an overview of existing methods for anomaly detection. Our proposed methodology and detailed framework is described in Section 3. Performance evaluation and analysis of experiment is followed in Section 4. Finally, Section 5 concludes the paper and sketches directions for possible future work.

2 RELATED WORK

Anomaly detection has been studied for decades. Based on whether the labels are used in the training process, they are grouped into supervised, semi-supervised and unsupervised anomaly detection. Our main focus is the unsupervised setting. In this section, we demonstrate various types of existing approaches for unsupervised anomaly detection, which can be categorized into traditional anomaly detection and deep anomaly detection.

2.1 Traditional anomaly detection

Conventional methods can be divided into three categories. 1) Reconstruction-based methods are proposed to represent and reconstruct accurately normal data by a model, for example, PCA [17], Kernel PCA [27], [28] and Robust PCA [29]. Specifically, RPCA is used to identify a low rank representation including random noise and outliers by using a convex relaxation of the rank operator; 2) Clustering analysis is used for anomaly detection, such as Gaussian Mixture Models (GMM) [30], k-means [18] and Kernel Density Estimator (KDE) [31]. They cluster different data samples and find anomalies via a predefined outlierness score; 3) the methods of one-class learning model are also widely used for anomaly detection. For instance, One-Class Support Vector Machine (OCSVM) [19] and Support Vector Data Description (SVDD) [32] seek to learn a discriminative hypersphere surrounding the normal samples and then classify new data as normal or abnormal.

It is notable that these conventional methods for anomaly detection are designed for static data. To capture the temporal dependencies appropriately, Autoregression (AR) [33], Autoregressive Moving Average (ARMA) [34] and Autoregressive Integrated Moving Average (ARIMA) model [35] are widely used. These models represent time series that are generated by passing the input through a linear or nonlinear filter which produces the output at any time using the previous output values. Once we have the forecast, we can use it to detect anomalies and compare with groundtruth. Nevertheless, AR model and its variants are rarely used in multi-sensor multivariate time series due to their high computational cost.

2.2 Deep anomaly detection

In deep learning-based anomaly detection, the reconstruction models, forecasting models as well as composite models will be discussed.

2.2.1 Reconstruction models

The reconstruction model focuses on reducing the expected reconstruction error by different methods. For instance, Autoencoders [20] are often utilized for anomaly detection by learning to reconstruct a given input. The model is trained exclusively on normal data. Once it is not able to reconstruct the input with equal quality compared to the reconstruction of normal data, the input sequence is treated as anomalous data. LSTM Encoder-Decoder model [36] is proposed to learn temporal representation of the input time series by LSTM networks and use reconstruction error to detect anomalies. Despite its effectiveness, LSTM does not

take spatial correlation into consideration. Convolutional Autoencoders (CAEs) [21] are an important method of video anomaly detection, which are able of capturing the 2D image structure since the weights are shared among all locations in the input image. Furthermore, since Convolutional long short-term memory (ConvLSTM) can model spatial-temporal correlations by using convolutional layers instead of fully connected layers, some researchers [15], [37] add ConvLSTM layers to autoencoder, which better encodes the change of appearance for normal data.

Variational Autoenocders (VAEs) are a special form of autoencoder that models the relationship between two random variables, latent variable z and visible variable x. A prior for z is usually multivariate unit Gaussian $\mathcal{N}(0, I)$. For anomaly detection, authors [38] define the reconstruction probability that is the average probability of the original data generating from the distribution. Data points with high reconstruction probability is classified as anomalies, vice versa. Others like Denoising AutoEncoders (DAEs) [22], Deep Belief Networks (DBNs) [39] and Robust Deep Autoencoder (RDA) [40] have also been reported good performance for anomaly detection.

2.2.2 Forecasting models

The forecasting model can also be used for anomaly detection. It aims to predict one or more continuous values, e.g. forecasting the current output values x_t for the past p values $[x_{t-p}, ..., x_{t-2}, x_{t-1}]$. RNN and LSTM is the standard model for sequence prediction. In the work [41], [42], authors perform anomaly detection by using RNN-based forecasting models to predict values for the next time period and minimize the mean squared error (MSE) between predicted and future values. Recently, there have also been attempted to perform anomaly detection using other feed-forward networks. For instance, Shalyga *et al.* [43] develop Neural Network (NN) based forecasting approach to early anomaly detection. Kravchik and Shabtai [44] apply different variants of convolutional and recurrent networks to perform forecasting model. And the results show that 1D convolutional networks obtain the best accuracy for anomaly detection in industrial control systems. In another work [45], Lai *et al.* propose a forecasting model, which uses CNN and RNN, namely LSTNet, to extract short-term local dependency pattern and long-term pattern for multivariate time series, and incorporates Linear SVR model in the LSTNet model. Besides, other efforts have been performed in [46] using GAN-based anomaly detection. The model adopts U-Net as generator to predict next frame in video and leverages the adversarial training to discriminate whether the prediction is real or fake, thus abnormal events can be easily identified by comparing the prediction and ground truth.

2.2.3 Composite models

Besides single model, composite model for unsupervised anomaly detection has gained a lot attention recently. Zong *et al.* [23] utilize a deep autoencoder to generate a low-dimensional representation and reconstruction error, which is further fed into a Gaussian Mixture Model to model density distribution of multi-dimensional feature. However, they cannot consider the spatial-temporal dependency for multivariate time series data. Different from this work, the

Composite LSTM model [47] uses single encoder LSTM and multiple decoder LSTMs to perform different tasks such as reconstructing the input sequence and predicting the future sequence. In [48], the authors use ConvLSTM model as a unit within the composite LSTM model following a branch for reconstruction and another for prediction. This type of composite model is currently used to extract features from video data for the tasks of action recognition. Similarly, authors in [49] propose Spatial-Temporal AutoEncoder (STAE) for video anomaly detection, which utilizes 3D convolutional architecture to capture the spatial-temporal changes. The architecture of the network is an encoder followed by two branches of decoder for reconstructing past sequence and predicting future sequence respectively.

As mentioned above, unsupervised anomaly detection techniques have still many deficiencies. For traditional anomaly detection, it is hard to learn representations of spatial-temporal patterns in multi-sensor time-series signals. For a reconstruction model, a single task could make the model suffer from the tendency to store information only about the inputs that are memorized by the AE. And for the forecasting model, this task could suffer from only storing the last few values that are most important for predicting the future [47], [48]. Hence, their performance will be limited since model only learn trivial representations. For composite model, these researchers design their models for different purposes. Zong *et al.* [23] could solve problem that the model is robust to noise and anomalies through performing density estimation in a low-dimensional space. Zhao *et al.* [49] could consider the spatial-temporal dependency through 3D convolutional reconstructing and forecasting architectures. However, few studies could address these issues simultaneously.

Different from these works, our research makes the following contributions: 1) The proposed model is designed to characterize complex spatial-temporal dependencies, thus discovering generalized pattern of multi-sensor data; 2) Adding a Maximum Mean Discrepancy (MMD) penalty could avoid the model generalizing so well for noisy data and anomalies; 3) Combining Attention-based Bidirectional LSTM (BiLSTM) and traditional Auto-regressive linear model could boost the model's performance from different time scale; 4) The composite baseline model is generated based on end-to-end training which means all the components within the model are jointly trained with compound objective function.

Besides, some learning algorithms based on time-series data have been studied for decades. [50] propose Unsupervised Salient Subsequence Learning to extract subsequence as new representations of the time series data. Due to the internally sequential relationship, many neural network-based models can be applied to time series in an unsupervised learning manner. For example, some 1D-CNN models [51], [52] have been proposed to solve time series tasks with a very simple structure and the sota performance. Moreover, the multiple time series signal usually has some kinds of co-relations, [53] propose a method to learn the relation graph on multiple time series. Some anomaly detection based on multiple time series applications are available for wastewater treatment [54], for ICU [55], and for sensors [56].
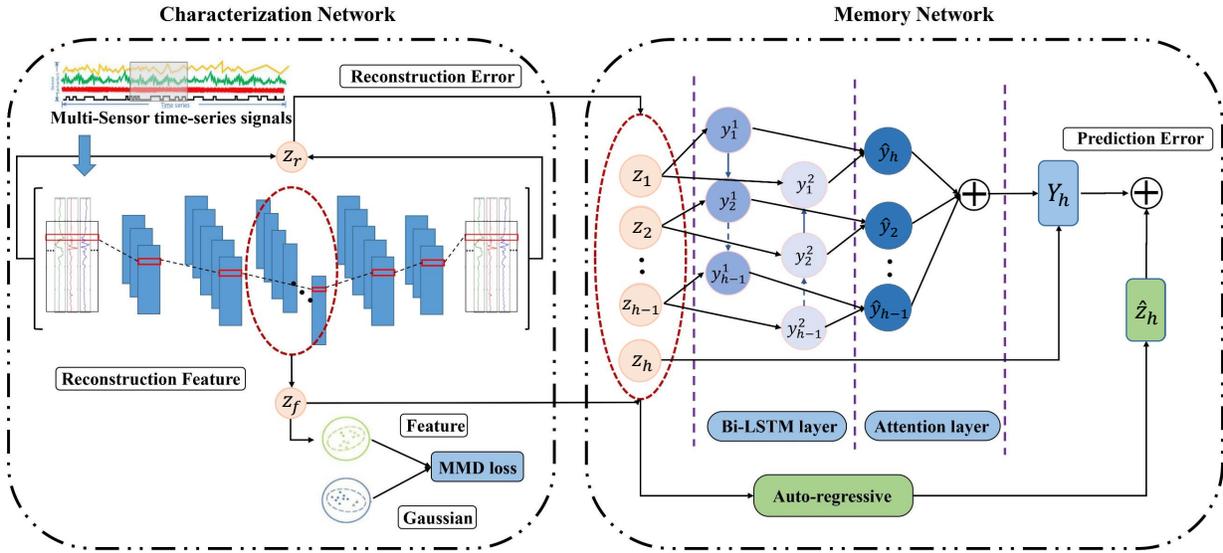
Fig. 1: The overview of the proposed CAE-M model.

3 THE PROPOSED METHOD

3.1 Notation

In a multi-sensor time series anomaly detection problem, we are given a dataset generated by n sensors ($n > 1$). Without loss of generality, we assume each sensor generates m signals (e.g., an accelerometer often generates 3-axis signals). Denote $\mathcal{S}$ the signal set, we have $N = |\mathcal{S}| = nm$ signals in total. For each signal $x_i \in \mathcal{S}$, $x_i \in \mathbb{R}^{t_i \times 1}$, where t_i denotes the length of signal x_i. Note that even each sensor signal may have different length, we are often interested in their intersections, i.e., all sensors are having the same length T, i.e., $X = (x_1, \cdots, x_N)^T \in \mathbb{R}^{N \times T}$ denotes an input sample containing all sensors.

Definition 1 (Unsupervised anomaly detection). It is nontrivial to formally define an anomaly. In this paper, we are interested in detecting anomalies in a classification problem. Let $\mathcal{Y} = \{1, 2, \cdots, K\}$ be the classification label set, and K the total number of classes, then the dataset $\mathcal{D} = (X_i, y_i)_{i=1}^N$. Eventually, our goal is to detect whether an input sample X_a belongs to one of the K predefined classes with a high confidence. If not, then we call X_a an anomaly. Note that in this paper, we are dealing with an unsupervised anomaly detection problem, where the labels are unseen during training, which is more obviously challenging.

3.2 Overview

There are some existing works [19], [21], [25] attempting to resolve the unsupervised anomaly detection problem. Unfortunately, they may face several critical challenges. First, conventional anomaly detection techniques such as PCA [17], k-means [18] and OCSVM [19] are unable to capture the temporal dependencies appropriately because they cannot deliver temporal memory states. Second, since the normal samples might contain noise and anomalies, using deep anomaly detection approaches such as standard Autoencoders [20], [21] is likely to affect the generalization capability. Third, the multi-stage approaches, i.e., feature

extraction and predictive model building are separated [25], [26], can easily get stuck in local optima.

In this paper, we present a novel approach called Convolutional Autoencoding Memory network (CAE-M) to tackle the above challenges. Fig. 1 gives an overview of the proposed method. In a nutshell, CAE-M is built upon a convolutional autoencoder, which is then fed into a predictive network. Concretely, we encode the spatial information in multi-sensor time-series signals into the low-dimensional representation via Deep Convolutional Autoencoder (CAE). In order to reduce the effect of noisy data, some existing works have tried to add Memory module [24] or Gaussian Mixture Model (GMM) [23]. In our proposed method, we simplify these modules into penalty item, which called Maximum Mean Discrepancy (MMD) penalty. Adding a MMD term can encourage the distribution of training data to approximate the same distribution such as Gaussian distribution, thus reducing the risk of overfitting caused by noise and anomalies in training data [23]. And then we feed the representation and reconstruction error to the subsequent prediction network based on Bidirectional LSTM (Bi-LSTM) with Attention mechanism and Auto-regressive model (AR) which could predict future feature values by modeling the temporal information. Through the composite model, the spatial-temporal dependencies of multi-sensor time-series signals can be captured. Finally, we propose a compound objective function with weight coefficients to guide end-to-end training. For normal data, the reconstructed value generated by data coding is similar to the original input sequence and the predicted value is similar to the future value of time series, while the reconstructed value and the predicted value generated by abnormal data change greatly. Therefore, in inference process, we can detect anomalies precisely by computing the loss function in composite model.

3.3 Characterization Network

In the characterization network, we perform representative learning by fusing multivariate signals in multiple sensors.

The low-dimensional representation contains two components: (1) the features which are abstracted from the multivariate signals; (2) the reconstruction error over the distance metrics such as Euclidean distance and Minkowski distance. To avoid the autoencoder generalizing so well for abnormal inputs, optimization function combines reconstruction loss by measuring how close the reconstructed input is to the original input and the regularization term by measuring the similarity between the two distributions (i.e., the distribution of low-dimensional features and Gaussian distribution).

3.3.1 Deep feature extraction

We employ a deep convolutional autoencoder to learn the low-dimensional features. Specifically, given N time series with length T, we pack into a matrix $x \in \mathbb{R}^{N \times T}$ with multi-sensor time-series data. The matrix is then fed to deep convolutional autoencoder (CAE). The CAE model is composed of two parts, an encoder and a decoder as in Eq. (1) and Eq. (2). Assuming that x' denotes the reconstruction of the same shape as x, the model is to compute low-dimensional representation z_f, as follows:

$$z_f = Encode(x), \tag{1}$$

$$x' = Decode(z_f). \tag{2}$$

The encoder in Eq. (1) maps an input matrix x to a hidden representation z_f by many convolutional and pooling layers. Each convolutional layer will be followed by a max-pooling layer to reduce the dimensions of the layers. A max-pooling layer pools features by taking the maximum value for each patch of the feature maps and produce the output feature map with reduced size according to the size of pooling kernel.

The decoder in Eq. (2) maps the hidden representation z_f back to the original input space as a reconstruction. In particular, a decoding operation needs to convert from a narrow representation to a wide reconstructed matrix, therefore the transposed convolution layers are used to increase the width and height of the layers. They work almost exactly the same as convolutional layers, but in reverse.

The difference between the original input vector x and the reconstruction x' is called the reconstruction error z_r. The error typically used in the autoencoder is Mean Squared Error (MSE), which measures how close the reconstructed input x' is to the original input x, as follows in Eq. (3).

$$L_{MSE} = \|x - x'\|_2^2, \tag{3}$$

where $\|\cdot\|_2^2$ is the l_2-norm.

3.3.2 Handling noisy data

To reduce the influence of noisy data, we need to observe the changes in low-dimensional features and the changes of distribution over the samples in a more granular way, thus distinguishing between normal and abnormal data obviously.

Inspired by [23], in order to avoid the autoencoder generalizing so well for noisy data and abnormal data, we hope to detect "lurking" anomalies that reside in low-density areas in the reduced low-dimensional space. Our proposed method is conceptually similar to Gaussian Mixture Model

(GMM) as target distributions. The loss function is complemented by MMD as a regularization term that encourages the distribution of the low-dimensional representation to be similar to a target distribution. It aims to make the distribution of noisy data close to the distribution of normal training data, thereby reducing the risk of overfitting. Specifically, Maximum Mean Discrepancy (MMD) [57] is a distance-measure between the samples of the distributions. Given the latent representation $z_a = \{z_f^{(1)}, ..., z_f^{(h)}\} \in \mathbb{R}^{h \times d}$, where d is a latent space (usually $d < N \times T$) and h denotes all of the time steps at one iteration. For CAE with MMD penalty, the Gaussian distribution P_z in reproduction kernel Hilbert space $\mathcal{H}$ is chosen as the target distribution. We compute the Kernel MMD as the follows:

$$L_{MMD}(Z, P_z) = \|\frac{1}{h}\sum_{i=1}^{h}\phi(z_f^{(i)}) - \frac{1}{h}\sum_{i=1}^{h}\phi(z^{(i)})\|_{\mathcal{H}}^2. \tag{4}$$

Here we have the distribution Z of the low-dimensional representation $z_f^{(i)}$ and the target distribution $z^{(i)} \sim P_z$ over a set $\mathcal{X}$. The MMD is defined by a feature map $\phi : \mathcal{X} \to \mathcal{H}$ where $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS).

During the training process, we could apply the kernel trick to compute the MMD. And it turns out that many kernels, including the Gaussian kernel, lead to the MMD being zero if and only the distributions are identical. Letting $k(x, y) = \langle\phi(x), \phi(y)\rangle_{\mathcal{H}}$, we yield an alternative characterization of the MMD as follows:

$$L_{MMD}(Z, P_z) = \|\frac{1}{h^2}\sum_{i \neq j}k(z_f^{(i)}, z_f^{(j)}) + \frac{1}{h^2}\sum_{i \neq j}k(z^{(i)}, z^{(j)})$$
$$- \frac{2}{h^2}\sum_{i,j}k(z_f^{(i)}, z^{(j)})\|_{\mathcal{H}}. \tag{5}$$

Here the kernel is defined as $k(u, v) = exp(-\frac{\|u-v\|^2}{2\sigma^2})$. The latent representation with Gaussian distribution P_z is performed by sampling from P_z and approximating by averaging the kernel $k(\cdot, \cdot)$ evaluated at all pairs of samples.

Note that we usually do batch training for neural network training. It means that the model is trained using a subsample of data at one iteration. In this work, we need to compute the MMD over a set of $\mathcal{X}$ at one iteration, where the number of $\mathcal{X}$ is equal to $batchsize \times timestep$. That is, the latent representation is denoted as $z_a = \{z_f^{(1)}, ..., z_f^{(l)}\} \in \mathbb{R}^{l \times d}$, where $l = batchsize \times h$.

3.4 Memory Network

To simultaneously capture the spatial and temporal dependencies, our proposed model is designed to characterize complex spatial-temporal patterns by concurrently performing the reconstruction analysis and prediction analysis. Considering the importance of temporal component in time series, we propose non-linear prediction and linear prediction to detect anomalies by comparing the future prediction and the next value appearance in the feature space.

The characterization network generates feature representations, which include reconstruction error and reduced low-dimensional features learned by the CAE at h time steps. Denote input features as z_h for $h = 1, ..., H$:

$$z_h = [z_f, z_r]_h, \ h \in [1, H]. \tag{6}$$

Our goal is to predict the current value z_h for the past values $[z_1, z_2, ..., z_{h-1}]$. The memory network combines non-linear function based predictor and linear function based predictor to tackle temporal dependency problem.

3.4.1 Non-linear prediction

Non-linear predictor function has different types such as Recurrent neural networks (RNNs), Long Short-Term Memory (LSTM) [58] and Gated Recurrent Unit (GRU) [59]. Original RNNs fall short of learning long-term dependencies. In this work, we adopt a Bidirectional LSTM with attention mechanism [60] which could consider the whole/local context while calculating the relevant hidden states. Specifically, the Bidirectional LSTM (BiLSTM) runs the input in two ways, one LSTM from past to future and one LSTM from future to past. Different from unidirectional, the two hidden states combined are able in any point in time to preserve information from both past and future. A BiLSTM unit consists of four components: input gate i_h, forget gate f_h, output gate o_h and cell activation vector c_h. The hidden state y_h given input z_h is computed as follows:

$$i_h = \sigma(W_{zi}z_h + W_{yi}y_{h-1} + W_{ci}c_{h-1} + b_i), \qquad (7)$$

$$f_h = \sigma(W_{zf}z_h + W_{yf}y_{h-1} + W_{cf}c_{h-1} + b_f), \qquad (8)$$

$$o_h = \sigma(W_{zo}z_h + W_{yo}y_{h-1} + W_{co}c_{h-1} + b_o), \qquad (9)$$

$$\widetilde{c}_h = tanh(W_{zc}z_h + W_{yc}h_{h-1} + W_{cc}c_{h-1} + b_c), \qquad (10)$$

$$c_h = f_h \otimes c_{h-1} + i_h \otimes \widetilde{c}_h, \qquad (11)$$

$$y_h = o_h \otimes tanh(c_h), \qquad (12)$$

$$\hat{y}_h = [y_h^1; y_h^2], \qquad (13)$$

where i_h, f_h, o_h, c_h represent the value of i, f, o, c at the moment h respectively, W and b denote the weight matrix and bias vector, $\sigma(\cdot)$ and $tanh(\cdot)$ are activation function, the operator $\otimes$ denotes element-wise multiplication, the current cell state c_h consists of two components, namely previous memory c_{h-1} and modulated new memory $\widetilde{c}_h$, the output $\hat{y}_h$ combines the forward y_h^1 and backward y_h^2 pass outputs. Note that the merge mode by which outputs of the forward and backward are combined has different types, e.g. sum, multiply, concatenate, average. In this work, we use the mode "sum" to obtain the output $\hat{y}_h$.

Attention mechanism for processing sequential data that could focus on the features of the keywords to reduce the impact of non-key temporal context. Hence, we adopt temporal attention mechanism to produce a weight vector and merge raw features from each time step into a segment-level feature vector, by multiplying the weight vector. The work process of attention mechanism is following detailed.

$$M_h = tanh(W_h\hat{y}_h + b_h), \qquad (14)$$

$$E_h = \sigma(W_a M_h + b_a), \qquad (15)$$

$$A_h = softmax(E_h), \qquad (16)$$

$$Y_h = \sum_h A_h * \hat{y}_h. \qquad (17)$$

Here W and b are represented as the weight and bias. A weighted sum of the $\hat{y}_h$ based on the weight A_h is computed as the context representation Y_h. The context representation is considered as the predicted value of z_h for temporal features $[z_1, z_2, ..., z_{h-1}]$.

3.4.2 Linear prediction

Autoregressive (AR) model is a regression model that uses the dependencies between an observation and a number if lagged observations. Non-linear Recurrent Networks are theoretically more expressive and powerful than AR models. In fact, AR models also yield good results in forecasting short term modeling. In specific real datasets, such infinite-horizon memory isn't always effective. Therefore, we incorporate AR model in parallel to the non-linear memory network part.

The AR model is formulated as follows:

$$\hat{z}_h = c \sum_{i=1}^{h-1} w_{h-i} + \sum_{i=1}^{h-1} w_{h-i} * z_{h-i}, \qquad (18)$$

where $w_1, ..., w_{h-1}$ are the weights of the AR model, c is a constant, $\hat{z}_h$ represents the predicted value for past temporal value $[z_1, z_2, ..., z_{h-1}]$. We implement this model using Dense layer of network to combine the weights and data.

In the output layer, the prediction error is obtained by computing the difference between the output of predictor model and true value z_h. The final prediction error integrates the output of non-linear prediction model and linear prediction model. The following equation is written as:

$$L_{predict} = \sum_{h \in \Omega_{batch}} (\underbrace{||Y_h - z_h||_F^2}_{\text{Attention-based BiLSTM}} + \underbrace{||\hat{z}_h - z_h||_F^2}_{\text{Autoregressive}}), \qquad (19)$$

where Ω_{batch} is a subsample of training data, $|| \cdot ||_F$ is the Frobenius norm.

3.5 Joint optimization

As for multi-step approach, it can easily get stuck in local optima, since models are trained separately. Therefore, we propose an end-to-end hybrid model by minimizing compound objective function.

The CAE-M objective has four components, MSE (reconstruction error) term, MMD (regularization) term, prediction error (non-linear forecasting task) term and prediction error (linear forecasting task) term. Given X samples $\{x_1, x_2, ..., x_D\}, x_i \in \mathbb{R}^{N \times T}$, the objective function is constructed as:

$$\begin{aligned} J(\theta) &= L_{MSE} + \lambda_1 \cdot L_{MMD} + \lambda_2 \cdot L_{lp} + \lambda_3 \cdot L_{np} \\ &= \frac{1}{M} \sum_{i=1}^{M} L(x_i, x_i') + \lambda_1 L_{MMD}(Z, P_Z) \\ &+ \frac{1}{M} \sum_{i=1}^{M} [\lambda_2 ||Y_h^{(i)} - z_h^{(i)}||_F^2 + \lambda_3 ||\hat{z}_h^{(i)} - z_h^{(i)}||_F^2], \end{aligned} \qquad (20)$$

where M is batch size used for training, h is current time step, λ_1, λ_2 and λ_3 are the meta parameters controlling the importance of the loss function.

Restating our goals more formally, we would like to:

- Minimize the reconstruction error in the characterization network, that is, minimize the error in reconstructing x' from x at all time step h. We need to compute the average error at each time step of sample. The purpose is to obtain better low-dimensional representation for multi-sensor data.

- Minimize the MMD loss that encourages the distribution Z of the low-dimensional representation to be similar to a target distribution P_z. It can make anomalies deviate from normal data in the reduced dimensions.

- Minimize the prediction error by integrating nonlinear predictor and linear predictor. We split the set $\{z_1, z_2, ..., z_h\}$ obtained by characterization network into the current value z_h and the past values $[z_1, z_2, ..., z_{h-1}]$. And then the predicted values Y_h and $\hat{z}_h$ are obtained by minimizing prediction errors. The purpose is to accurately express the information of the next temporal slice using different predictor, thus updating low-dimensional feature and reconstruction error.

- λ_1, λ_2 and λ_3 are the meta parameters in CAE-M. In practice, $\lambda_1 = e - 04$, $\lambda_2 = 0.5$, and $\lambda_3 = 0.5$ usually achieve desirable results. Here MMD is complemented as a regularization term. The parameter selection is performed in Section 4.8.1.

3.6 Inference

Given samples as training dataset $X = \{x_1, x_2, ..., x_D\}, x_i \in \mathbb{R}^{N \times T}$, we are able to compute the corresponding decision threshold (THR):

$$\text{THR} = \frac{1}{D} \sum_{i=1}^{D} \text{Err}(x_i) + \sqrt{\frac{1}{D} \sum_{i=1}^{D} (\text{Err}(x_i) - \mu)^2}, \quad (21)$$

where we denote $\text{Err}(x_i)$ as the sum of loss function for x_i, and μ is the average value of $\text{Err}(x_i)$ for $i = 1, ..., D$. The setting is similar to the normal training distribution $\mathcal{N}(\mu, \sigma)$ following with 1 standard deviation σ of the mean μ.

In inference process, the decision rule is that if $\text{Err}(x_i) >$ THR, the testing sample in a sequence can be predicted to be "abnormal", otherwise "normal".

The complete training and inference procedure of CAE-M is shown in Algorithm 1.

4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the performance of our proposed CAE-M approach for anomaly detection on several real-world datasets.

4.1 Datasets

We adopt two large publicly-available datasets and a private dataset: PAMAP2, CAP and Mental fatigue dataset. These datasets are exploiting multi-sensor time series for activity recognition, sleep state detection, and mental fatigue detection, respectively. Therefore, they are ideal testbeds for evaluating anomaly detection algorithms.

PAMAP2 [61] dataset is a mobile dataset with respect to actions or activities from UCI repository, containing data of 18 different physical activities performed by 9 subjects wearing 3 inertial measurement units, e.g. accelerator, gyroscope and magnetometer. There are 18 activity categories in total. For experiments, we treat these classes with relatively

---

Algorithm 1 Training and Inference procedure of CAE-M

Training process

Input: Normal Dataset $X = \{x_1, x_2, ..., x_D\}$, time steps h, batch size M and hyperparameters $\lambda_1, \lambda_2, \lambda_3$.

Output: Anomaly decision threshold THR and model parameter w.

1: Transform each sample $x \in \mathbb{R}^{N \times T}$ into $x \in \mathbb{R}^{h \times N \times t}$ in the time axis;
2: Randomly initialize parameter w;
3: **while** not converge **do**
4:     Calculate low-dimensional representation z_f and reconstruction error z_r at each time step; // Eq. (1) (3)
5:     Calculate MMD between z_a and Gaussian distribution P_z; // Eq. (5)
6:     Combine z_f and z_r into $z_h = [z_f, z_r]_h$ for each sample; // Eq. (6)
7:     Predict the current value z_h for the past values $[z_1, z_2, ..., z_{h-1}]$ by Attention-based BiLSTM and AR model; // Eq. (7-18)
8:     Update w by minimizing the compound objective function; // Eq. (20)
9: **end while**
10: Calculate the decision threshold THR by the training samples; // Eq.(21)
11: **return** Optimal w and THR.

Inference process

Input: Normal and Anomalous dataset $X = \{x_1, x_2, ..., x_D\}$, threshold THR, model parameter w, hyperparameters λ_1, λ_2 and λ_3.

Output: Label of all x_i.

1: **for all** x_i **do**
2:     Calculate the loss $Err(x_i) = f(x_i; w)$; // $f(\cdot)$ denotes CAE-M
3:     **if** $Err(x_i) >$ THR **then**
4:         x_i = "anomaly";
5:     **else**
6:         x_i = "normal";
7:     **end if**
8: **end for**
9: **return** Label of all x_i.

---

smaller samples as the anomaly classes (including running, ascending stairs, descending stairs and rope jumping), while the rest categories are combined to form the normal classes.

CAP Sleep Database [62], which stands for the Cyclic Alternating Pattern (CAP) database, is a clinical dataset from PhysioNet repository. It is characterized by periodic physiological signals occurring during wake, S1-S4 sleep stages and REM sleep. The waveforms include at least 3 EEG channels, 2 EOG channels, EMG signal, respiration signal and EKG signal. There are 16 healthy subjects and 92 patients in the database. The pathological recordings include the patients diagnosed with bruxism, insomnia, narcolepsy, nocturnal frontal lobe epilepsy, periodic leg movements, REM behavior disorder and sleep-disordered breathing. In this task, we extracted 7 valid channels of all the channels like ROC-LOC, C4-P4, C4-A1, F4-C4, P4-O2, ECG1-ECG2, EMG1-EMG2 etc. For detecting sleep apnea

TABLE 1: The detailed statistics of three datasets

Dataset	Domain	Instances	Dimensions	Classes	Permissions
PAMAP2 [61]	Activity Recognition	1,140,000	27	18	Public
CAP [62]	Sleep Stage Detection	921,700,000	21	8	Public
Mental Fatigue Dataset [2]	Fatigue Detection	1,458,648	4	2	Private

events, we chose healthy subjects as normal class and the patients with sleep-disordered breathing as anomaly class.

Mental Fatigue Dataset [2] is a real world health-care dataset. Aiming to detect mental fatigue in the healthy group, we collected the physiological signals (e.g., GSR, HR, R-R intervals and skin temperature) using wearable device. There are 6 healthy young subjects participated in the mental fatigue experiments. In this task, non-fatigue data samples are labeled as normal class and fatigue data samples are labeled as anomaly class. Fatigue data accounts for a fifth of the total.

The detailed information of the datasets is shown in TABLE 1.

4.2 Baseline Methods

In order to extensively evaluate the performance of the proposed CAE-M approach, we compare it with several traditional and deep anomaly detection methods:

(1) **KPCA** (Kernel principal component analysis) [28], which is a non-linear extension of PCA commonly used for anomaly detection. (2) **ABOD** (Angle-based outlier detection) [63], which is a probabilistic model that well suited for high dimensional data. (3) **OCSVM** (One-class support vector machine) [64], which is the one-class learning method that classifies new data as similar or different to the training set. (4) **HMM** (Hidden Markov Model) [65] is a finite set of states, each of which is associated with a probability distribution. In a particular state an observation can be generated, according to the associated probability distribution. (5) **CNN-LSTM** [66], which is a forecasting model composed of convolutional and LSTM networks. It can obtain the forecast by estimating the current data, and detect anomalies on comparing the forecasting value with actuals. (6) **LSTM-AE** (LSTM based autoencoder) [36], which is an unsupervised detection technique used in time series that can induce a representation by learning an approximation of the identity function of data. (7) **ConvLSTM-COMPOSITE** [48], which utilizes a composite structure that is able to encoder the input, reconstruct it, and predict its near future. To simplify the name, "ConvLSTM-COMP" denotes ConvLSTM-COMPOSITE. We choose the "conditional" version to build a single model called **ConvLSTM-AE** by removing the forecasting decoder. (8) **UODA** (Unsupervised sequential outlier detection with deep architecture) [67], which utilizes autoencoders to capture the intrinsic difference between normal and abnormal samples, and then integrates the model to RNNs that perform fine-tuning to update the parameters in DAE. (9) **MSCRED** (Multi-scale convolutional recurrent encoder-decoder) [15], which is a reconstruction-based anomaly detection and diagnosis method.

4.3 Implementation details

For traditional anomaly detection, we scale the sequential data into segments and extract the features from each segment. In PAMAP2 dataset, multiple sensors are worn on three different position (wrist, chest, ankle). Hence, we extract 324 features including time and frequency domain features. In CAP Sleep dataset, we first pass through the Hanning window low pass filter for removing the high frequency components of signals. And then we extract 91 features for EEG, EMG and ECG signals [68]–[70]; In Mental Fatigue dataset, we preprocess physiological signals by interpolation and filtering algorithm. Then we extract 23 features for Galvanic Skin Response (GSR), Heart Rate (HR), R-R intervals and skin temperature sensors [2].

For Deep Anomaly Detection (DAD) method, we filter multi-sensor signals and then pack these signals into matrix as input to construct the deep model.

We reimplement these methods based on open-source repositories[1] or our own implementations. For KPCA, we employ Gaussian kernel with a bandwidth of 600, 500, 0.5, respectively for PAMAP2, CAP, and Mental Fatigue datasets. For ABOD, we use k nearest neighbors to approximate the complexity reduction. For an observation, the variance of its weighted cosine scores to all neighbors could be viewed as the abnormal score. For OCSVM, we adopt PCA for OCSVM as a dimension reduction tool and employ the Gaussian kernel with a bandwidth of 0.1. For HMM, we build a Markov model after extracting features and calculate the anomaly probability from the state sequence generated by the model. For CNN-LSTM, we define a CNN-LSTM model in `Keras` by first defining 2D convolutional network as comprised of Conv2D and MaxPooling2D layers ordered into a stack of the required depth, wrapping them in a TimeDistributed layer and then defining the LSTM and output layers. For LSTM-AE, we use single-layer LSTM on both encoder and decoder in the task. For ConvLSTM-COMPOSITE, we choose "conditional" version and adapt this technique to anomaly detection in multivariate time series. Here we also build a single model called ConvLSTM-AE by removing forecasting decoder. For UODA, we reimplement this algorithm by customizing the number of layers and hyper-parameters. For MSCRED, we first construct multi-scale matrices for multi-sensor data, and then fed it into MSCRED model and evaluate the performance.

For our own CAE-M, we use library Hyperopt [71] to select the best hyper-parameters (i.e., time window, the number of neurons, learning rate, activation function, optimization criteria and iterations). The characterization network runs with $Conv2D \rightarrow Maxpooling \rightarrow Conv2D \rightarrow Maxpooling \rightarrow Conv2DTranspose \rightarrow Conv2DTranspose \rightarrow Conv2DTranspose$, i.e., Conv1-Conv5 with 32 kernels of size 4 × 4, 64 kernels of size 4 × 4, 64 kernels of size 4 × 4, 32 kernels of size 4 × 4, 1 kernels of size 4 × 4, and Maxpooling with size 2 × 2. We use Rectified Linear Unit (ReLU) as the activation function of convolutional

---

1. https://pyod.readthedocs.io/en/stable/, https://github.com/7fantasysz/MSCRED

TABLE 2: The mean precision, recall and F1 score of baselines and our proposed method, * p-value = 0.0077.

Method	PAMAP2			CAP dataset			Fatigue dataset		
	mPre	mRec	mF1	mPre	mRec	mF1	mPre	mRec	mF1
KPCA	0.7236	0.6579	0.6892	0.7603	0.5847	0.6611	0.5341	0.5014	0.5173
ABOD	0.8653	0.9022	0.8834	0.7867	0.6365	0.7037	0.6679	0.6145	0.6401
OCSVM	0.7600	0.7204	0.7397	0.9267	0.9259	0.9263	0.5605	0.5710	0.5290
HMM	0.6950	0.6553	0.6745	0.8238	0.8078	0.8157	0.6066	0.6076	0.6071
CNN-LSTM	0.6680	0.5392	0.5968	0.6159	0.5217	0.5649	0.5780	0.5042	0.5386
LSTM-AE	0.8619	0.7997	0.8296	0.7147	0.6253	0.6671	0.7140	0.6820	0.6870
UODA	0.8957	0.8513	0.8730	0.7557	0.5124	0.6107	0.8280	0.7770	0.8017
MSCRED	0.6997	0.7301	0.7146	0.6410	0.5784	0.6081	0.8016	0.6802	0.7359
ConvLSTM-AE	0.7359	0.7361	0.7360	0.8150	0.8194	0.8172	0.9010	0.9346	0.9175
ConvLSTM-COMP	0.8844	0.8842	0.8843	0.8367	0.8377	0.8372	0.9373	0.9316	0.9344
CAE-M (Ours)	**0.9608**	**0.9670**	**0.9639**	**0.9939**	**0.9952**	**0.9961**	**0.9962**	**0.9959**	**0.9960**
Improvement	7.64%	6.48%	7.96%	6.72%	6.93%	6.98%	5.89%	6.13%	6.16%

layers. The memory network contains non-linear prediction and linear prediction, where the non-linear network runs with $BiLSTM(512) \rightarrow Attention(h-1) \rightarrow Dropout(0.2) \rightarrow FC(1000, linear)$, and the linear network runs with $FC(1000, linear)$. The CAE-M model is trained in an end-to-end fashion using `Keras` [72]. The optimization algorithm is Adam and the batch size is set as 32. And we set parameters of compound objective function $\lambda_1 = e - 04$, $\lambda_2 = 0.5$ and $\lambda_3 = 0.5$. The time step h usually gives desirable results as $h = 5$ or $h = 10$.

Note that in addition to the complete CAE-M approach, we further evaluate its several variants as baselines to justify the effectiveness of each component:

- CAE-M*w/o*Pre. The CAE-M model removes the linear and non-linear prediction. That is, this variant only adopts the characterization network with reconstruction loss and MMD loss. (i.e., $\lambda_1 = e - 04, \lambda_2 = 0, \lambda_3 = 0$)

- CAE-M*w/o*Rec+MMD. The CAE-M model removes the reconstruction error and MMD. Different from CNN-LSTM model, the characterization network is still performed as the deep convolutional autoencoder. We put the latent representation without reconstruction error into the memory network. (i.e., $\lambda_1 = 0, \lambda_2 = 0.5, \lambda_3 = 0.5$)

- CAE-M*w/o*ATTENTION. The CAE-M model without Attention component is implemented. (i.e., $\lambda_1 = e - 04, \lambda_2 = 0.5, \lambda_3 = 0.5$)

- CAE-M*w/o*AR. The CAE-M model without AR component is implemented. (i.e., $\lambda_1 = e - 04, \lambda_2 = 0.5, \lambda_3 = 0$)

- CAE-M*w/o*MMD. The CAE-M model without MMD component is implemented. (i.e., $\lambda_1 = 0, \lambda_2 = 0.5, \lambda_3 = 0.5$)

Note that anomaly detection problems are often with highly-imbalanced classes, hence *accuracy* is not suitable as the evaluation metric. In order to thoroughly evaluate the performance of our proposed method, we follow existing works [15], [23], [73] to adopt the mean *precision, recall*, and *F1 score* as the evaluation metrics. The mean *precision* means the average precision of normal and abnormal class. The same pattern goes for mean *recall, F1* score.

In the experiments, the train-validation-test sets are split by following existing works [15], [67]. Concretely speaking, for each dataset, we split normal samples into training, validation, and test with the ratio of $5 : 1 : 4$, where the training and validation set only contain normal samples and have no overlapping with testing set. The anomalous samples are only used in the testing set. The model selection criterion, i.e., hyperparameters, used for tuning is the validation error on the validation set.

4.4 Results and Analysis

As shown in TABLE 2, we compare our proposed method with traditional and deep anomaly detection methods using the mean precision, recall and F1 score. We can see that our method outperforms most of the existing methods, which demonstrates the effectiveness of our method. From TABLE 2, we can observe the following results.

For the PAMAP2 dataset, the CAE-M achieves the highest precision and recall compared by 10 popular methods. Traditional methods perform differently on PAMAP2 dataset since they are limited by the feature extraction and feature selection methods. In deep learning method, CNN-LSTM has a lowest F1 score. This means that more constraints such as data preprocessing method and anomaly evaluation strategy need to be added for prediction-based anomaly detection. For LSTM-AE, MS-CRED and ConvLSTM-AE, they both are reconstruction-based anomaly detection methods. Their performance is limited by the "noisy data" problem, resulting in reconstruction error for the abnormal input could be fit so well. For UODA, it performs reasonably well on the PAMAP2 dataset, but it is not end-to-end training, which is needed by pre-training denoising autoencoder (DAEs) and deep recurrent networks (RNNs), and then fine-tuning the UODA model composing of the DAE and RNN. For ConvLSTM-COMPOSITE model, it performs better than other baseline models. The model consists of a single encoder, two decoders of reconstruction branch and prediction branch. In

TABLE 3: Detection performance in different sleep stages of baselines and our proposed method, *p-value = 0.0074.

Method	WAKE			S1			S2			S3			S4			REM		
	mPre	mRec	mF1	mPre	mRec	mF1	mPre	mRec	mF1	mPre	mRec	mF1	mPre	mRec	mF1	mPre	mRec	mF1
KPCA	0.9162	0.8213	0.8662	0.8267	0.7598	0.7918	0.9257	0.9353	0.9305	0.9039	0.8689	0.8861	0.9402	0.9604	0.9502	0.9536	0.9614	0.9575
ABOD	0.9872	0.8686	0.9242	0.9347	0.5522	0.6942	0.9389	0.6550	0.7716	0.8489	0.6184	0.7155	0.6749	0.6448	0.6595	0.5915	0.5909	0.5912
OCSVM	0.9784	0.9492	0.9636	0.9655	0.9504	0.9579	0.9395	0.9448	0.9421	**0.9714**	**0.9499**	**0.9605**	0.8701	0.9488	0.9077	**0.9784**	0.9492	0.9636
HMM	0.8417	0.8406	0.8411	0.8790	0.8856	0.8823	0.8967	0.8887	0.8927	0.6880	0.6747	0.6813	0.7279	0.7286	0.7282	0.8024	0.8649	0.8325
LSTM-AE	0.6990	0.7178	0.7082	0.6517	0.6492	0.6504	0.7430	0.7331	0.7380	0.7689	0.7828	0.7758	0.7274	0.7569	0.7418	0.6590	0.6887	0.6735
UODA	0.6159	0.6326	0.6241	0.6762	0.6762	0.6762	0.7290	0.5223	0.6086	0.5716	0.5766	0.5741	0.6626	0.8498	0.6807	0.5626	0.6116	0.5861
ConvLSTM-COMP	0.9889	0.9772	0.9830	0.9755	0.9850	0.9864	0.9250	0.9127	0.9188	0.9401	0.9041	0.9217	0.8647	0.8866	0.9023	0.9675	0.9949	0.9810
CAE-M	**0.9974**	**0.9949**	**0.9961**	**0.9958**	**0.9950**	**ß0.9954**	**0.9950**	**0.9950**	**0.9950**	0.9294	0.8842	0.9063	**0.9842**	**0.9950**	**0.9895**	0.9681	**0.9950**	**0.9813**

fact, since its efficiency is influenced by reconstruction error and prediction error respectively, its performance could be limited by one of encoder-decoder models.

For the CAP dataset, most of methods show a low F1 score. As CAP dataset contains different sleep stages of subjects, some methods are limited by high complexity of data. For OCSVM and HMM, they achieve better performance because of dimensionality reduction from 36 dimensions of PAMAP2 dataset to 7 dimensions. For MSCRED, due to *batch size =1* for the training model in the open source code, the loss function couldn't converge during training model and the training speed is slow. Our proposed method achieves about 7% improvement at F1 score, compared with the existing methods.

For Fatigue dataset, it is difficult to label fatigue and non-fatigue data manually. Therefore, it may be a lot of noise or misclassification patterns in the data, so that most of methods fail to solve this problem. For UODA, MSCRED and ConvLSTM, they have ability to overcome noise and misclassification of training data. Our proposed method also solves this problem successfully and achieves at least 6% improvement at F1 score.

Besides, in order to indicate significant differences from our proposed method and other baselines, we use Wilcoxon signed rank test [74] to analyze these results in TABLE 2. We compute average p-value of CAE-M compared with other baselines. A p-value = 0.0077 indicates that the performance of our proposed method differs from other methods. This p-value is also computed in TABLE 3.

4.5 Fine-grained Analysis

In addition to the anomaly detection of different classes on each dataset, we conduct a fine-grained analysis to evaluate the performance of each method within each class. Considering intra-class diversity, we conduct a group of experiments to detect anomalies in different sleep stages. In fact, these physiological signals in different sleep stages have significant differences. We choose 4 traditional methods and 3 deep methods with good performance in global domain as comparison methods. As shown in TABLE 3, we can observe that our architecture is most robust across same experiment settings. Several observations from these results are worth highlighting. For ABOD, the testing performance is unstable in local domain, which the highest F1 score is 0.92 in WAKE and the lowest F1 score is 0.59 in REM. For KPCA and

TABLE 4: The evaluation results on LOSO cross validation approach, including the best, the worst and the mean F1 score of 8 subjects.

Method	Worst mF1	Best mF1	Mean mF1
ABOD	0.6093	0.8507	0.7706
ConvLSTM-COMP	0.7033	0.9224	0.8493
UODA	0.5938	0.9336	0.7984
CAE-M	**0.8009**	**0.9433**	**0.8616**

ConvLSTM-COMPOSITE, the testing performance in local domain far exceeds the performance in global domain. This demonstrates that the two model can achieve better performance when intra-class data have similar distribution or regular pattern. For other methods, the testing performance is consistent in local and global domain. For our proposed method, the best testing performance can be achieved no matter in local domain or global domain. This study clearly justifies the superior representational capacity of our architecture to solve intra-class diversity.

4.6 Effectiveness Evaluation

4.6.1 Leave One Subject Out

In this section, we measure the generalization ability of models using Leave One Subject Out (LOSO). The fact is that when training and testing datasets contain the same subject, the model is likely to know more about the current subject which may be biased towards a new one. Therefore, LOSO could help to evaluate the generalization ability. We choose the PAMAP2 dataset to conduct subject-independent experiments which contain 8 subjects. As can be seen in Fig. 2(a), we evaluate our proposed method and three methods with relatively high F1 score. By examining the results, one can easily notice that deep learning-based methods obtain better performance than traditional methods. However, complex models such as deep neural networks are prone to overfitting because of their flexibility in memorizing the idiosyncratic patterns in the training set, instead of generalizing to unseen data.

TABLE 4 shows the best, the worst and average performance among 8 subjects. We can observe that UODA and ConvLSTM-COMPOSITE model perform well in some specific subjects, but they fail to reduce the effects of overfitting to each test subject, even drop to 0.70 and 0.59

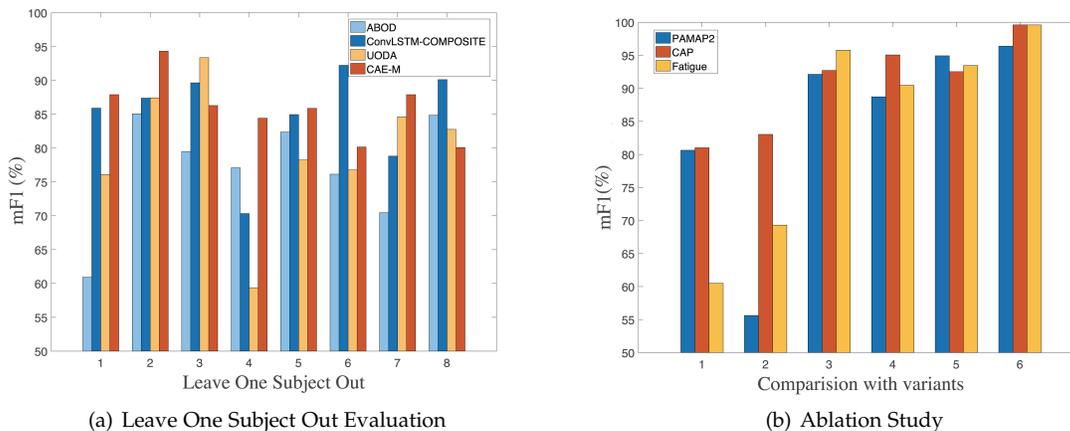(a) Leave One Subject Out Evaluation (b) Ablation Study

Fig. 2: Effectiveness evaluation using LOSO method and ablation study.

TABLE 5: The repeated measures analysis of variance on LOSO cross validation approach of one subject.

Method	mPre	mRec	mF1
ABOD	0.6240±0.000	0.5946±0.000	0.6090±0.000
ConvLSTM-COMP	0.8953±0.029	0.8081±0.038	0.8488±0.019
UODA	0.8155±0.063	0.7464±0.027	0.7782±0.031
CAE-M	**0.9437±0.024**	**0.8191±0.003**	**0.8770±0.012**

(a) mF1 (b) mPre (c) mRec

Fig. 3: Robustness to noisy data.

for some subjects. Compared to these methods, CAE-M can generalize well on testing subjects it hasn't appeared before, which reach the average F1 score of 0.86. Besides, we perform an analysis of variance on repeated measures within subject 1 (corresponding to numbers in Fig. 2(a)). As shown in TABLE 5, we observe that CAE-M remains a more stable performance on repeated measurements. In summary, the above demonstrates that our model can be motivated to improve the generalization ability.

4.6.2 Ablation Study

The proposed CAE-M approach consists of several components such as CAE, MMD, Attention mechanism, BiLSTM and Auto-regressive. To demonstrate the effectiveness of each component, we conduct ablation studies in this section. The ablation study is shown in Fig. 2(b). These ID numbers represent CAE-M without non-linear and linear prediction, CAE-M without reconstruction error and MMD, CAE-M without attention module, CAE-M without AR, CAE-M without MMD and CAE-M, respectively. The experimental results indicate that for the removal of different component above, there is corresponding performance drop at F1 score. We can observe that CAE-M model without prediction or reconstruction error achieves a low F1 score relatively. This demonstrates that our composite model is effective and necessary for anomaly detection in multi-sensor time-series data. Compared to original CAE-M model, removing the AR component (in CAE-M$_{w/o\ AR}$) from the full model causes significant performance drops on most of the datasets. This shows the critical role of the AR component in general. Moreover, attention and MMD components can also cause big performance rises on all the datasets. More details are
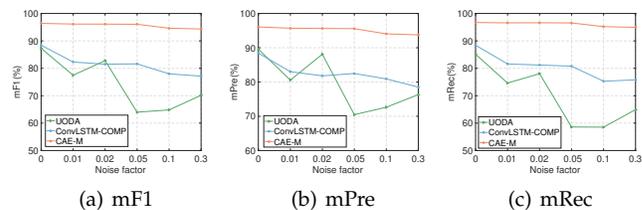
shown in TABLE 6. Here, these ID numbers are corresponding to numbers in Fig. 2(b).

4.7 Robustness to Noisy Data

In real-world applications, the collection of multi-sensor time-series data can be easily polluted with noise due to changes in the environment or the data collection devices. The noisy data bring critical challenges to the unsupervised anomaly detection methods. In this section, we evaluate the robustness of different methods to noisy data. We manually control the noisy data ratio in the training data. We inject Gaussian noise (μ=0, σ=0.3) in a random selection of samples with a ratio varying between 1% to 30%. We compare the performance of three methods on PAMAP2 dataset: UODA, ConvLSTM-COMPOSITE, and CAE-M in Fig. 3. These methods have good stability in the above experiments. As the noise increases, the performance of all methods decreases. For CAE-M, the F1 score, precision and recall have no significant decline. Among them, our model remains significantly superior to others, demonstrating its robustness to noisy data.

4.8 Further Analysis

4.8.1 Parameter Sensitivity Analysis

In this section, we evaluate the parameter sensitivity of CAE-M model. It is worth noting that CAE-M achieves the best performance by adjusting weight coefficient of compound objective function. We apply control variate reduction technique [75] to empirically evaluate the sensitivity

TABLE 6: The mean precision, recall and F1 score from variants.

ID	Method	PAMAP2			CAP dataset			Fatigue dataset		
		mPre	mRec	mF1	mPre	mRec	mF1	mPre	mRec	mF1
1	CAE-M$_{w/o\ Pre}$	0.8103	0.8023	0.8063	0.8299	0.8101	0.8199	0.6005	0.6096	0.6050
2	CAE-M$_{w/o\ Rec+MMD}$	0.5693	0.5440	0.5563	0.8896	0.7784	0.8303	0.7050	0.6814	0.6930
3	CAE-M$_{w/o\ ATTENTION}$	0.9151	0.9276	0.9213	0.9251	0.9291	0.9271	0.9605	0.9551	0.9578
4	CAE-M$_{w/o\ AR}$	0.9060	0.8691	0.8872	0.9634	0.9381	0.9506	0.9046	0.9048	0.9047
5	CAE-M$_{w/o\ MMD}$	0.9437	0.9550	0.9493	0.9293	0.9213	0.9253	0.9407	0.9288	0.9347
6	CAE-M	**0.9608**	**0.9670**	**0.9639**	**0.9939**	**0.9952**	**0.9961**	**0.9962**	**0.9959**	**0.9960**

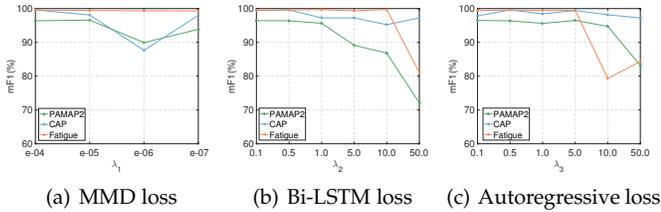(a) MMD loss   (b) Bi-LSTM loss   (c) Autoregressive loss

Fig. 4: Parameter sensitivity analysis of the proposed CAE-M approach.

of parameter $\lambda_1, \lambda_2, \lambda_3$ with a wide range. The results are shown in Fig. 4. As the value of MMD loss is greater than others, we select its weight coefficient within e-04 $\sim$ e-07 and other weight coefficients within [0.1, 0.5, 1, 5, 10, 50]. We adjust one of λ while fixing the other respective λ to keep the optimal value ($\lambda_1 = e - 04$, $\lambda_2 = 0.5$, and $\lambda_3 = 0.5$). When weight coefficient is increased, we observe that F1 score tends to decline. The optimal parameter is $\lambda_1 = e - 04$, $\lambda_2 = 0.5$, and $\lambda_3 = 0.5$. It can be seen that the performance of CAE-M stays robust within a wide range of parameter choice.
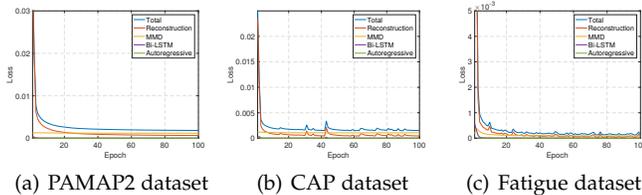

(a) PAMAP2 dataset   (b) CAP dataset   (c) Fatigue dataset

Fig. 5: Convergence analysis of the proposed CAE-M approach on different datasets.

4.8.2 Convergence Analysis

Since CAE-M involves several components, it is natural to ask whether and how quickly it can converge. In this section, we analyze the convergence to answer this question. We extensively show the results of each component on three datasets in Fig. 5. These results demonstrate that even if the proposed CAE-M approach involves several components, it could reach a steady performance within fewer than 40 iterations. Therefore, in real applications, CAE-M can be applied more easily with a fast and steady convergence performance.

5 CONCLUSION AND FUTURE WORK

In this paper, we introduced a Deep Convolutional Autoencoding Memory network named CAE-M to detect anomalies. The CAE-M model uses a composite framework to model generalized pattern of normal data by capturing spatial-temporal correlation in multi-sensor time-series data. We first build Deep Convolutional Autoencoder with a Maximum Mean Discrepancy (MMD) penalty to characterize multi-sensor time-series signals and reduce the risk of overfitting caused by noise and anomalies in training data. To better represent temporal dependency of sequential data, we use non-linear Bidirectional LSTM with Attention and linear Auto-regressive model for prediction. Extensive empirical studies on HAR and HC datasets demonstrate that CAE-M performs better than other baseline methods.

In the future work, we will focus on the point-based fine-grained anomaly detection approach and further improve our method for multi-sensor data by designing proper sparse operations.

REFERENCES

[1] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[2] Y. Zhang, Y. Chen, and Z. Pan, "A deep temporal model for mental fatigue detection," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 1879–1884.

[3] J. E. Ball, D. T. Anderson, and C. S. Chan, "Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community," *Journal of Applied Remote Sensing*, vol. 11, no. 4, p. 042609, 2017.

[4] B. Kiran, D. Thomas, and R. Parakkal, "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos," *Journal of Imaging*, vol. 4, no. 2, p. 36, 2018.

[5] A. Ramchandran and A. K. Sangaiah, "Unsupervised anomaly detection for high dimensional data—an exploratory analysis," in *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*. Elsevier, 2018, pp. 233–251.

[6] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.

[7] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K. Müller, "A unifying review of deep and shallow anomaly detection," 2020.

[8] Y. Zhang, Y. Chen, and C. Gao, "Deep unsupervised multi-modal fusion network for detecting driver distraction," *Neurocomputing*, vol. 421, pp. 26–38, 2021.

[9] Y. Chen, J. Wang, M. Huang, and H. Yu, "Cross-position activity recognition with stratified transfer learning," *Pervasive and Mobile Computing*, vol. 57, pp. 1–13, 2019.

[10] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[11] J. Wang, V. W. Zheng, Y. Chen, and M. Huang, "Deep transfer learning for cross-domain activity recognition," in *proceedings of the 3rd International Conference on Crowd Science and Engineering*, 2018, pp. 1–8.

[12] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[13] J. Wang, Y. Chen, L. Hu, X. Peng, and S. Y. Philip, "Stratified transfer learning for cross-domain activity recognition," in *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2018, pp. 1–10.

[14] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," in *International Conference on Artificial Neural Networks*. Springer, 2019, pp. 703–716.

[15] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1409–1416.

[16] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," *arXiv preprint arXiv:1511.06309*, 2015.

[17] R. Paffenroth, P. Du Toit, R. Nong, L. Scharf, A. P. Jayasumana, and V. Bandara, "Space-time signal processing for distributed pattern detection in sensor networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 38–49, 2013.

[18] L. J. Latecki, A. Lazarevic, and D. Pokrajac, "Outlier detection with kernel density functions," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*. Springer, 2007, pp. 61–75.

[19] 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.

[20] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. ACM, 2014, p. 4.

[21] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 733–742.

[22] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[23] 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.

[24] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. v. d. Hengel, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," *arXiv preprint arXiv:1904.02639*, 2019.

[25] D. Li, D. Chen, J. Goh, and S.-k. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," *arXiv preprint arXiv:1809.04758*, 2018.

[26] R. Aliakbarisani, A. Ghasemi, and S. F. Wu, "A data-driven metric learning-based scheme for unsupervised network anomaly detection," *Computers & Electrical Engineering*, vol. 73, pp. 71–83, 2019.

[27] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[28] H. Hoffmann, "Kernel pca for novelty detection," *Pattern recognition*, vol. 40, no. 3, pp. 863–874, 2007.

[29] R. Paffenroth, K. Kay, and L. Servi, "Robust pca for anomaly detection in cyber networks," *arXiv preprint arXiv:1801.01571*, 2018.

[30] R. Laxhammar, G. Falkman, and E. Sviestins, "Anomaly detection in sea traffic-a comparison of the gaussian mixture model and the kernel density estimator," in *2009 12th International Conference on Information Fusion*. IEEE, 2009, pp. 756–763.

[31] J. Kim and C. D. Scott, "Robust kernel density estimation," *Journal of Machine Learning Research*, vol. 13, no. Sep, pp. 2529–2565, 2012.

[32] D. M. Tax and R. P. Duin, "Support vector data description," *Machine learning*, vol. 54, no. 1, pp. 45–66, 2004.

[33] N. Günnemann, S. Günnemann, and C. Faloutsos, "Robust multivariate autoregression for anomaly detection in dynamic product ratings," in *Proceedings of the 23rd international conference on World wide web*. ACM, 2014, pp. 361–372.

[34] J. D. Hamilton, *Time series analysis*. Princeton university press Princeton, NJ, 1994, vol. 2.

[35] H. Z. Moayedi and M. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," in *2008 International Symposium on Information Technology*, vol. 4. IEEE, 2008, pp. 1–6.

[36] 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.

[37] W. Luo, W. Liu, and S. Gao, "Remembering history with convolutional lstm for anomaly detection," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2017, pp. 439–444.

[38] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, no. 1, 2015.

[39] D. Wulsin, J. Blanco, R. Mani, and B. Litt, "Semi-supervised anomaly detection for eeg waveforms using deep belief nets," in *2010 Ninth International Conference on Machine Learning and Applications*. IEEE, 2010, pp. 436–441.

[40] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 665–674.

[41] P. Filonov, F. Kitashov, and A. Lavrentyev, "Rnn-based early cyber-attack detection for the tennessee eastman process," *arXiv preprint arXiv:1709.02232*, 2017.

[42] T. Ergen, A. H. Mirza, and S. S. Kozat, "Unsupervised and semi-supervised anomaly detection with lstm neural networks," *arXiv preprint arXiv:1710.09207*, 2017.

[43] D. Shalyga, P. Filonov, and A. Lavrentyev, "Anomaly detection for water treatment system based on neural network with automatic architecture optimization," *arXiv preprint arXiv:1807.07282*, 2018.

[44] M. Kravchik and A. Shabtai, "Detecting cyber attacks in industrial control systems using convolutional neural networks," in *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*. ACM, 2018, pp. 72–83.

[45] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 95–104.

[46] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection–a new baseline," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6536–6545.

[47] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International conference on machine learning*, 2015, pp. 843–852.

[48] J. R. Medel and A. Savakis, "Anomaly detection in video using predictive convolutional long short-term memory networks," *arXiv preprint arXiv:1612.00390*, 2016.

[49] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, and X.-S. Hua, "Spatio-temporal autoencoder for video anomaly detection," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 1933–1941.

[50] Q. Zhang, J. Wu, P. Zhang, G. Long, and C. Zhang, "Salient subsequence learning for time series clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 9, pp. 2193–2207, 2018.

[51] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean, "Inceptiontime: Finding alexnet for time series classification," *Data Mining and Knowledge Discovery*, vol. 34, no. 6, pp. 1936–1962, 2020.

[52] W. Tang, G. Long, L. Liu, T. Zhou, J. Jiang, and M. Blumenstein, "Rethinking 1d-cnn for time series classification: A stronger baseline," *arXiv preprint arXiv:2002.10061*, 2020.

[53] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 753–763.

[54] B. Jun, "Fault detection using dynamic time warping (dtw) algorithm and discriminant analysis for swine wastewater treatment," *Journal of hazardous materials*, vol. 185, no. 1, pp. 262–268, 2011.

[55] W. Chen, S. Wang, G. Long, L. Yao, Q. Z. Sheng, and X. Li, "Dynamic illness severity prediction via multi-task rnns for intensive care unit," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 917–922.

[56] X. Zhang, L. Yao, C. Huang, S. Wang, M. Tan, G. Long, and C. Wang, "Multi-modality sensor data classification with selective attention," *arXiv preprint arXiv:1804.05493*, 2018.

[57] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A hilbert space embedding for distributions," in *International Conference on Algorithmic Learning Theory*. Springer, 2007, pp. 13–31.

[58] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[59] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[60] G. Liu and J. Guo, "Bidirectional lstm with attention mechanism and convolutional layer for text classification," *Neurocomputing*, vol. 337, pp. 325–338, 2019.

[61] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *2012 16th International Symposium on Wearable Computers*. IEEE, 2012, pp. 108–109.

[62] M. G. Terzano, L. Parrino, A. Smerieri, R. Chervin, S. Chokroverty, C. Guilleminault, M. Hirshkowitz, M. Mahowald, H. Moldofsky, A. Rosa *et al.*, "Atlas, rules, and recording techniques for the scoring of cyclic alternating pattern (cap) in human sleep," *Sleep medicine*, vol. 3, no. 2, pp. 187–199, 2002.

[63] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 444–452.

[64] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Proceedings of the International Joint Conference on Neural Networks, 2003.*, vol. 3. IEEE, 2003, pp. 1741–1745.

[65] S. S. Joshi and V. V. Phoha, "Investigating hidden markov models capabilities in anomaly detection," in *Proceedings of the 43rd annual Southeast regional conference-Volume 1*. ACM, 2005, pp. 98–103.

[66] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[67] W. Lu, Y. Cheng, C. Xiao, S. Chang, S. Huang, B. Liang, and T. Huang, "Unsupervised sequential outlier detection with deep architectures," *IEEE transactions on image processing*, vol. 26, no. 9, pp. 4321–4330, 2017.

[68] D. P. Subha, P. K. Joseph, R. Acharya, and C. M. Lim, "Eeg signal analysis: a survey," *Journal of medical systems*, vol. 34, no. 2, pp. 195–212, 2010.

[69] A. Phinyomark, C. Limsakul, and P. Phukpattaranont, "A novel feature extraction for robust emg pattern recognition," *arXiv preprint arXiv:0912.3973*, 2009.

[70] M. K. Gautama and V. K. Giri, "An overview of feature extraction techniques of ecg," *American-Eurasian Journal of Scientific Research*, vol. 12, no. 1, pp. 54–60, 2017.

[71] J. Bergstra, D. Yamins, and D. D. Cox, "Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms," in *Proceedings of the 12th Python in science conference*. Citeseer, 2013, pp. 13–20.

[72] N. Ketkar, "Introduction to keras," in *Deep learning with Python*. Springer, 2017, pp. 97–111.

[73] Y. Guo, W. Liao, Q. Wang, L. Yu, T. Ji, and P. Li, "Multidimensional time series anomaly detection: A gru-based gaussian mixture variational autoencoder approach," in *Asian Conference on Machine Learning*, 2018, pp. 97–112.

[74] D. Rey and M. Neuhäuser, "Wilcoxon-signed-rank test," *Springer Berlin Heidelberg*.

[75] S. Kucherenko, B. Delpuech, B. Iooss, and S. Tarantola, "Application of the control variate technique to estimation of total sensitivity indices," *Reliability Engineering & System Safety*, vol. 134, pp. 251–259, 2015.