Curriculum Learning-Based Multivariate Time Series Anomaly Detection

Devilliers Caleb Dube, Mehmet Akar
Boğaziçi University
Istanbul, Turkey
devilliers.dube@std.bogazici.edu.tr, mehmet.akar@bogazici.edu.tr

***Abstract*—Anomaly detection in multivariate time series (MVTS) is a significant research domain across several industries, including cybersecurity and industrial systems. There have been several deep learning-based methods proposed within this research domain. The development of high capacity frameworks has received the majority of attention in recent years due to the availability of large open-source datasets and improvement in computer processing power. However, there has not been much attention in investigating the importance of how the data is presented to these techniques during training. Curriculum learning (CL), a technique based on ordered learning, was proposed for machine learning. In CL, the model first learns from easy data and is progressively trained with increasingly difficult data. In this paper, we propose data-based CL for MVTS anomaly detection. We further introduce the CL concept to the learner (model), in which we first train a simple model and then utilize a complex model in the final training round. To the best of our knowledge, we are the first to investigate these approaches in MVTS anomaly detection. We evaluate the proposed designs on the SWaT dataset using the F1 score and the results show an improvement in performance.**

I. Introduction

Anomaly detection has been a prominent research area for several years. Previously, data being generated by sensors in the industry was of relatively small dimension, not easily accessible, and in some cases, simply univariate. Univariate time series is a sequence of data of a single variable collected over a period of time. Currently, massive amounts of data with large dimension are constantly being generated by several sensors in cyberphysical systems. Since they contain multiple variables, they are referred to as *multivariate time series* (MVTS). These variables may exhibit complex temporal correlation making it very difficult to identify a single or series of data points which deviate from the distribution of the rest of the data. Such data points are known as anomalies and can be identified via *anomaly detection*. Anomaly detection is mainly implemented in industrial systems [1], cybersecurity [2], fraud detection [3], network intrusion detection [4], and has been introduced to other areas such as textile processes [5].

Several anomaly detection methods such as statistical, machine learning, and deep learning have been proposed across different domains [6]. Statistical or probabilistic models typically regard anomalies as statistical outliers or extreme values. Additionally, a few statistical tests may be used to identify these data points. The primary advantage of these techniques is that they are easily applicable to any particular type of data. However, selecting the wrong probability distribution could result in poor anomaly detection performance. A seasonal autoregressive integrated moving average (SARIMA) technique is proposed in [7]. SARIMA was trained and evaluated on a small network traffic data. This conventional technique requires a lot of training time and may fail learn to when subjected to large MVTS data.

Machine learning and deep learning frameworks can be classified into *supervised*, *semi-supervised*, and *unsupervised* methods [6]. In supervised anomaly detection, the training set includes both labeled normal and anomalous data points. In semi-supervised techniques, only the normal data is used for training whereas no labeled data is used in unsupervised frameworks. It is generally difficult to label the data, hence supervised techniques are not preferred. Machine learning techniques such as k-means clustering [8] and support vector machines [9] can be used for anomaly detection. However, these techniques may fail to identify periodic, seasonality patterns, and correlations within MVTS data.

Curriculum learning (CL) is a machine learning approach that involves training a model on a sequence of data samples in a specific order, starting with simpler data and advancing to more complex ones. It addresses the problems of catastrophic forgetting, enhances convergence rate of the training process, and improves model performance in some tasks [10]. The concept of CL for machine learning was introduced in [11] and has since gained interest across several fields such as in federated face recognition [12] and natural language understanding [13] tasks. Prior research as summarized in [14] has shown that the benefits of CL cannot be generalized across all fields. Therefore, it is necessary to conduct research in new areas with various curriculum designs to analyze the domain-specific effects of CL.

In MVTS anomaly detection, to our knowledge, there has been no research which conducted extensive experiments with CL. In this paper, we train a semi-supervised reconstruction-based deep learning technique using CL. The contributions of our work are summarized as follows:

- We introduce the CL concept to MVTS anomaly detection and perform several experiments to determine the efficacy of CL-trained frameworks.
- We design data-based CL techniques for MVTS anomaly detection. Furthermore, we analyze which data feature can be selected for best model performance improvement.

- We show that data-based curricula can improve performance of the model-based framework.

The rest of the paper is structured as follows: First, the related work section presents some pertinent methods in the literature. Subsequently, the methodology section defines the problem and offers a comprehensive description of the proposed approach. Then, the experiments section provides details of the dataset, experimental setup, and results. Lastly, the conclusion summarizes the principal findings of this work and outlines the direction for future research.

II. Related Work

Deep learning-based techniques have proven to be effective in learning important features in high dimensional MVTS data, hence they dominate the state-of-the-art (SOTA) techniques [15]. Deep learning anomaly detection techniques can be classified into three main categories, namely, *reconstruction-based*, *forecasting-based*, and *generative* methods [15].

Reconstruction-based techniques mainly employ an encoder to learn the spatial information from normal data only, a decoder to reconstruct the data, and reconstruction errors as anomaly scores. The fundamental principle is that the model should sufficiently learn only the normal patterns and poorly reconstruct the previously unseen anomalous instances. In [16], EncDecAD, an LSTM-based encoder-decoder framework is proposed and the results show that it can detect predictable and unpredictable anomalies. MSCRED [17] introduces the concept of representing the data through signature matrices to MVTS anomaly detection. An adversarially trained transformer based approach is proposed in [18]. The use of adversarial training enables it to improve its decoding ability.

Forecasting-based methods typically perform training on normal historical data, forecast future values or patterns, and then use the deviations as anomaly scores. If a forecasting horizon contains anomalies, it is expected to produce a high prediction error. In [19], the authors used stacked LSTM layers for learning temporal patterns and represented the forecasting errors as a Gaussian distribution. DeepAnT [20] was trained on unlabeled data to learn the distribution for forecasting the normal patterns of the data. It utilizes a predictor based on convolutional neural networks (CNNs) to make forecasts for the next data point and an anomaly detector to identify the points which deviate from the distribution.

Generative techniques use a model to learn the distribution of the data and then classify anomalies as data points with probability below a defined marginal likelihood. These techniques utilize a generative model such as Gaussian mixture model, variational autoencoder, and generative adversarial networks. In DAGMM [21], MVTS anomaly detection was performed without considering the temporal dependencies within the data. The researchers fed only a single observation with several features to the model rather than temporal patterns. In MVTS, learning temporal patterns is very important, hence their method may offer substandard performance in some datasets. Other proposed generative techniques are STAD-GAN [22] and MadSGM [23].

III. Methodology

In this section, we provide a brief description of the Multi-Scale Convolutional Recurrent Encoder-Decoder (MSCRED) framework which was designed for anomaly detection and root cause diagnosis [17]. Then, we explain how we design CL-based frameworks for anomaly detection.

A. Problem Statement

MVTS anomaly detection aims at identifying anomalies that exist in the data which, for instance, may be collected from sensors in industrial systems. Due to the large dimension of MVTS data, conducting anomaly detection is more difficult in comparison to in univariate time series data.

Consider an n-dimension MVTS training data of length L; $X_{train} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n) \in \mathbb{R}^{L \times n}$ where $\mathbf{x}_i \in \mathbb{R}^L$. The algorithm has to learn only the normal temporal information for it to identify novel patterns in the test set as anomalies. Therefore, the training data should not consist of any anomaly. For anomaly detection, an output vector $\mathbf{y} \in \mathbb{R}^L$ that consists of either "0" for normal or "1" for anomaly has to be produced.

We propose to solve this problem using CL-based frameworks. CL is based on identifying training criteria which constitute a curriculum [11] that can be used for progressive training of the algorithm. A curriculum, $C = \langle Q_1, \ldots, Q_k, \ldots, Q_K \rangle$ with training data X_{train} divided into K subsets is a sequence of training criteria Q_k which is a reweighted version of the targeted probability distribution $P(\mathbf{x})$ and is expressed as:

$$Q_k(\mathbf{x}) \propto W_k(\mathbf{x}) P(\mathbf{x}), \quad \forall \mathbf{x} \in X_{train} \tag{1}$$

where $W_k(\mathbf{x})$ is the weight and the following conditions have to be satisfied:

i) $H(Q_k) < H(Q_{k+1})$, where $H(Q_k)$ is the entropy
ii) $W_k(\mathbf{x}) \leq W_{k+1}(\mathbf{x})$
iii) $Q_K(\mathbf{x}) = P(\mathbf{x})$

According to the first condition, the entropy should increase in proportion to the diversity and information of the selected training criterion. As training continues, the likelihood of sampling more difficult training data increases. Since more data is being added gradually, under the second condition, the weight should likewise increase. The third condition indicates that all the training data is utilized in the final training round [14]. Moreover, we propose to extend the CL principle to the models. We suggest to use the algorithmic complexity of different anomaly detection models as the curriculum for the model-based CL technique.

B. Overview of MSCRED

MSCRED is a reconstruction-based technique which uses: i) system signature matrices (SSM) [24] to represent the data, ii) fully convolutional encoder [25] for encoding spatial patterns of the SSM, iii) an attention-based Convolutional Long Short-Term Memory (ConvLSTM) recurrent neural network [26] to model temporal patterns, and iv) convolutional decoder for reconstructing the SSM.

The concept behind MSCRED is that previously unseen temporal patterns during the encoding stage are poorly reconstructed, hence, are labeled as anomalies. The researchers in [24] proposed a Toeplitz Inverse Covariance-Based Clustering technique for MVTS data where each cluster in their method is represented by a correlation network. They observed that in addition to temporal dynamics in MVTS, the correlations between the time series pairs are essential for characterizing the status of the system. Given two time series in the time windows from $t-\omega$ to t obtained from the i-th and j-th sensors, respectively; $\mathbf{x}_i^\omega = [x_{i,t-\omega+1}, x_{i,t-\omega+2}, \cdots, x_{i,t}]^{\mathrm{T}} \in \mathbb{R}^\omega$, and $\mathbf{x}_j^\omega = [x_{j,t-\omega+1}, x_{j,t-\omega+2}, \cdots, x_{j,t}]^{\mathrm{T}} \in \mathbb{R}^\omega$, their inner product and correlation matrices are obtained respectively as:

$$m_{ij}^t = \frac{\left\langle \mathbf{x}_i^\omega, \mathbf{x}_j^\omega \right\rangle}{f} \tag{2}$$

$$M^t = \begin{bmatrix} m_{11}^t & m_{12}^t & \cdots & m_{1n}^t \\ m_{21}^t & m_{22}^t & \cdots & m_{2n}^t \\ \vdots & \vdots & \vdots & \vdots \\ m_{n1}^t & m_{n2}^t & \cdots & m_{nn}^t \end{bmatrix} \tag{3}$$

where $m_{ij}^t \in M^t$ and f is a scaling factor, $f = \omega$ in this framework. Moreover, three window sizes are used to represent short, medium, and long inter-sensor correlations. Then, the SSM are obtained by concatenating M^t with these window sizes to represent multi-scale inter-sensor correlations. SSM are a mathematical representation of the data that provide a summary of the correlations and patterns within the data.

Subsequently, these SSM are fed to the convolutional encoder to encode spatial patterns. Each of the data layers is represented as a three-dimensional array with dimensions of $n \times n \times d$, where $n \times n$ represents spatial dimensions and d is the channel dimension [25]. Following that, an attention-based ConvLSTM is used to capture the temporal information within the data. The attention mechanism allows the ConvLSTM cell to adaptively select the most important hidden states for learning the data features. During the update, not all previous steps are essential for the current state, thus, the temporal attention mechanism adaptively selects the most relevant prior states for temporal pattern modeling. The SSM are then reconstructed using a convolutional decoder. In reconstruction, a reverse order is followed by feeding the last hidden state of the ConvLSTM layer to a convolutional decoder designed to obtain the output with the same size as that of the input SSM. The following square loss function is used to assess the model learning with the objective of minimizing reconstruction errors:

$$\mathcal{L} = \sum_{c=1}^{s} \left\| \mathcal{X}_{:,:,c}^t - \hat{\mathcal{X}}_{:,:,c}^t \right\|_2^2, t = 1, 2, ... \tag{4}$$

where $\mathcal{X}_{:,:,c}^t$ and $\hat{\mathcal{X}}_{:,:,c}^t$ denote channel c of the input and reconstructed SSM, $\mathcal{X}^t$ and $\hat{\mathcal{X}}^t$ respectively. The residual matrices which are obtained by calculating the difference between the input and reconstructed SSM are used for anomaly detection. The number of inaccurately reconstructed entries of the SSM is used as the anomaly scores. A threshold that maximizes the anomaly detection F1 score is determined experimentally. This framework is adopted in this work and improved using CL as explained in the following subsection.

C. Curriculum Design

In order to establish a CL design, there are three fundamental questions that should be addressed:

1) What criteria are utilized to order the training data according to difficulty?
2) What criteria are utilized to order the models according to increasing algorithmic complexity?
3) For the data-based method, when should the algorithm be presented with more difficult data?

By establishing two basic components, the aforementioned questions may be addressed. To answer the first two questions, it is necessary to establish a "Difficulty Measurer" that will allow the data or models to be sorted from easy to complex. For the third question, in order to sample the training data to be fed to the algorithm at each round, a "Training Scheduler" must be established. The training scheduler selects which data to feed the algorithm until all of the data is sampled. Once these questions are answered, there need to be a method for producing the previously described components which facilitate CL [14]. The proposed data-based and model-based CL approaches are shown in Fig. 1 and Fig. 2, respectively. Furthermore, Algorithms 1 and 2 which are inspired by the Baby Step algorithm [11], summarize the data-based and model-based CL methods, respectively. In our framework, we manually create the training scheduler and difficulty measurer using a criterion known as predefined CL which requires prior knowledge of the features of the data [27] or model. In this research, we experiment with different difficulty measures and training schedulers as follows:

1) Data-Based Curriculum Design Based on Sequence Length: We use the sequence length of the data as the difficulty measure. We draw motivation from the observation that traditional machine learning frameworks struggle to achieve good generalization when provided with large datasets. In deep learning, there are three main factors that have contributed to performance improvement: i) increased capacity of the models, ii) higher computational capability, and iii) the accessibility of massive amounts of data. The authors in [28] investigated the efficacy of the data in deep learning. They performed computer vision experiments and observed that large data improves representation learning. This implies that substantial acquisition of a larger-scale dataset can significantly enhance deep leaning model performance. Furthermore, a study in [29] presents experiments with small data and the results are significantly lower than those of high-capacity models with large data. Therefore, the size of the dataset has an effect in model learning and performance, hence, it can be used to design a curriculum. We split the data percentage-wise and feed it to the MSCRED framework with gradual increase in data length as follows:

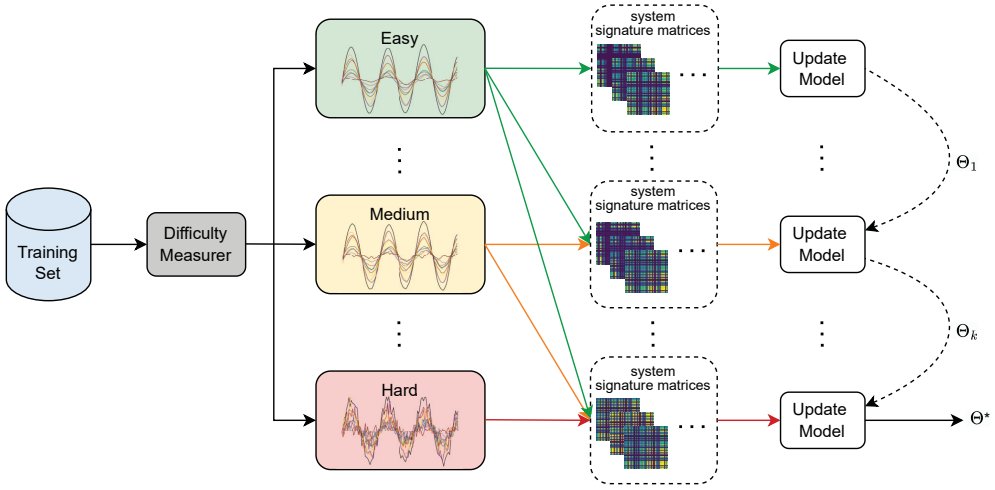

Fig. 1. Proposed data-based curriculum learning for multivariate time series anomaly detection framework. The system signature matrices of the easy subset are used in the first training round. Then, the model parameters from this training are used as the initial parameters for the following training round, and the experiments are repeated with the easy signature matrices augmented with more difficult ones.

Algorithm 1 Data-based curriculum learning for multivariate time series anomaly detection

Inputs: X_{train}: Training set; X_{test}: Test set; X_{CL}: Curriculum training set; C: Difficulty measurer; th: Anomaly threshold.
Outputs: Θ^*: Optimal model; A: Anomaly score.
$X'_{train} = \text{sort}(X_{train}, C)$;
$\{X^1_{train}, X^2_{train}, \cdots, X^K_{train}\} = X'_{train}$ where $C(\mathbf{x}_a) < C(\mathbf{x}_b),\ \mathbf{x}_a \in X^i_{train},\ \mathbf{x}_b \in X^j_{train}, \forall i < j$;
$X_{CL} = \emptyset$;
for $k = 1 \cdots K$ **do**
  $X_{CL} = X_{CL} \cup X^k_{train}$;
  Generate system signature matrices from X_{CL};
  while not converged for E epochs **do**
    train(Θ, X_{CL});
  end while
end for
Generate system signature matrices from X_{test};
for test data points $x_{test} \in X_{test}$ **do**
  test(x_{test}, Θ^*);
  if $A(x_{test}) > th$ **then**
    x_{test} is an anomaly;
  end if
end for

- **DCL-MSCRED-P2**: Data-based CL for MSCRED based on the dual percentage-wise split. The MSCRED model is first trained with 50% of the data. Then, it is trained with the whole data in the last training round.
- **DCL-MSCRED-P3**: Data-based CL for MSCRED based on the ternary percentage-wise split. The MSCRED model is first trained with about 33% of the data, followed by using 67% of the data. Finally, it is trained with the whole data in the last training round.

2) Data-Based Curriculum Design Based on Window Size: In this approach, we propose to utilize the window size of the data as the difficulty measure. The window size refers to the number of data points used as input features to reconstruct the next data point in reconstruction-based methods. We define a shorter window size to be easy because it is easier to reconstruct the data that is in close proximity to the first point in the window. However, as the window size increases, data variability, trend, and seasonality patterns may change significantly making it more difficult to reconstruct the data. We propose the following designs:

- **DCL-MSCRED-W2**: Data-based CL for MSCRED based on two different window sizes. The MSCRED model is first trained to learn the short window, $\omega = 10$ data points (100 seconds). Then, it is trained to learn the short and long windows, $\omega = 10, 60$.
- **DCL-MSCRED-W3**: Data-based CL for MSCRED based on three different window sizes. The MSCRED model is first trained to learn the short window, $\omega = 10$. Then, the medium window is included, $\omega = 10, 30$. Finally, the algorithm is trained to learn the short, medium, and long windows, $\omega = 10, 30, 60$.

3) Model-Based Design Based on Algorithmic Complexity: In this technique, we propose to apply CL to the learner, i.e., the model. Deep learning models vary in architecture, model size, and optimization process. We seek to investigate the effect of applying CL based on the architectural complexity of the model. We propose the following CL-based framework:

- **MCLAD**: Model-based CL for Anomaly Detection. Comparing model complexity for deep learning frameworks is a challenging task especially if the models are designed from very diverse neural networks [30]. We adopt the LSTM-ED [31] which has an architecture closely related to MSCRED. Since MSCRED consists of CNNs and an attention mechanism in addition to the

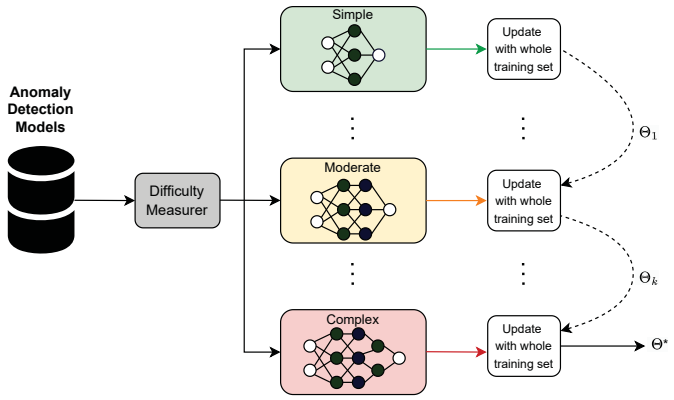

Fig. 2. Proposed model-based curriculum learning for multivariate time series anomaly detection framework. First, the simple model is updated using signature matrices of the whole training data. Subsequently, this data is introduced to a more complex model for training using the previously trained model as the initial model for each successive training round.

Algorithm 2 Model-based CL for multivariate time series anomaly detection

Inputs: X_{train}: Training set; X_{test}: Test set; C: Difficulty measurer; $\{\Theta^a, \Theta^b, \cdots, \Theta^z\}$: Anomaly detection models ordered in increasing complexity, $C(\Theta^a) < C(\Theta^b) < , \cdots, < C(\Theta^z)$; th: Anomaly threshold.
Outputs: Θ^*: Optimal model; A: Anomaly score.
Generate system signature matrices from X_{train};
for each model $\Theta \in \{\Theta^a, \Theta^b, \cdots, \Theta^z\}$ **do**
  while not converged for E epochs **do**
    train(Θ, X_{train});
  end while
  Set Θ as initial model for subsequent training round;
end for
Generate system signature matrices from X_{test};
for test data points $x_{test} \in X_{test}$ **do**
  test(x_{test}, Θ^*);
  if $A(x_{test}) > th$ **then**
    x_{test} is an anomaly;
  end if
end for

LSTM in its architecture, we use it as the more complex model. Moreover, we separately evaluate these models with the expectation that the high capacity model will outperform the low capacity one when trained with the whole training data. We train the LSTM-ED model with the SSM obtained from the whole dataset and transfer the model parameters to be the initial parameters of the MSCRED which is also trained with all the training data.

4) Data and Model-Based Curriculum Design: In [29], besides observing the importance of large data for high capacity models, it was also discovered that relatively less complex models perform better with small data than their more complex counterparts. From this observation, we draw inspiration to combine the data-based and model-based CL frameworks. We propose to train the less complex model with small data and transfer the learned model parameters to be the initial parameters for the training of the more complex model with the whole data. We design the following curriculum:

- **D-MCLAD**: Data and Model-based CL for Anomaly Detection. The LSTM-ED framework is trained with 50% of the data. Then, the MSCRED is trained with the whole training data and we perform evaluation on the test data as shown in Fig. 3.

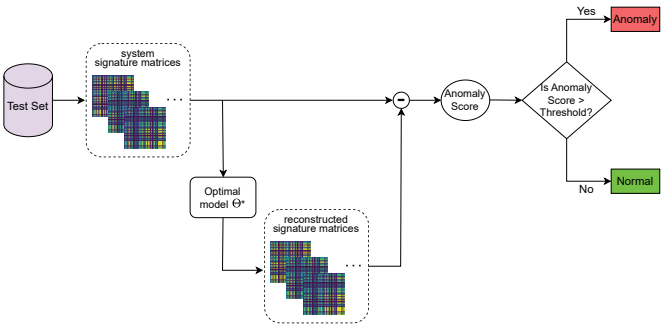

Fig. 3. Reconstruction-based anomaly detection

IV. Experiments

A. Dataset

The Secure Water Treatment (SWaT) dataset is a cyber-security dataset collected from a real-world water treatment testbed and is widely used in MVTS anomaly detection. The SWaT testbed which consists of six stages, is a small version of a real industrial water treatment facility. Its setup includes programmable logic controllers (PLCs) and a supervisory control and data acquisition (SCADA) system among other systems. The dataset includes a variety of data types that were obtained from the operational technology network of the plant, such as sensor readings, actuator statuses, network traffic, and system logs. It also consists of two communication networks: level 0 which allows each PLC to communicate with its sensors and actuators and level 1 which facilitates communication between the PLCs and SCADA.

The purpose of this dataset is to aid research in industrial control system (ICS) security by understanding the effects of anomalies in ICSs and evaluating the extent of cyber and physical attacks on ICSs. In addition, it is used to assess the performance of anomaly detection frameworks and determine the efficiency of defense mechanisms. In this research, we utilize the dataset which contains just the physical attacks. The data was collected over a period of 11 days in which the first 7 days were under normal operation and attacks were launched in the remaining 4 days. The data points of the first 30 minutes were removed to prevent starting process instability as the systems require a certain amount of time for startup before stabilizing. The dataset we used contains a total of $946,722$ samples, representing 51 time series. The duration of the attacks vary from about one minute to an hour [32], [33]. Fig. 4 shows some of the time series data and the anomaly locations in the SWaT dataset.

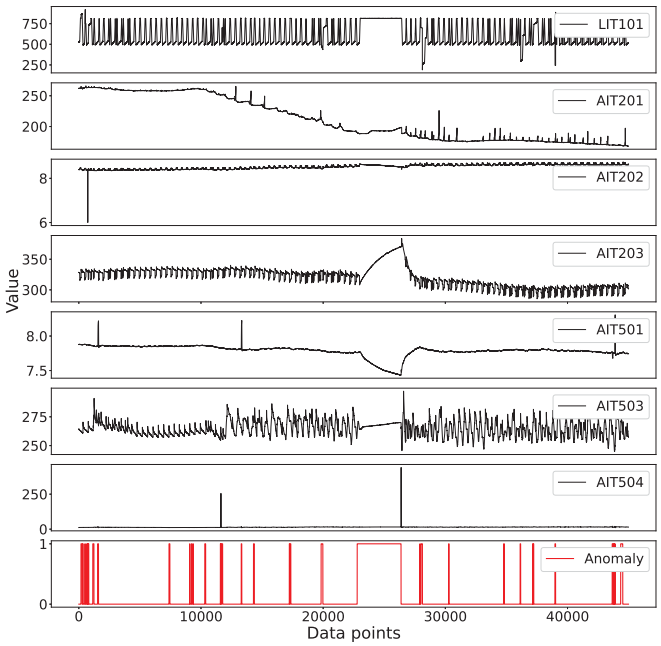

Fig. 4. Time series data of some of the sensors of the downsampled SWaT test set with the respective labeled anomalies

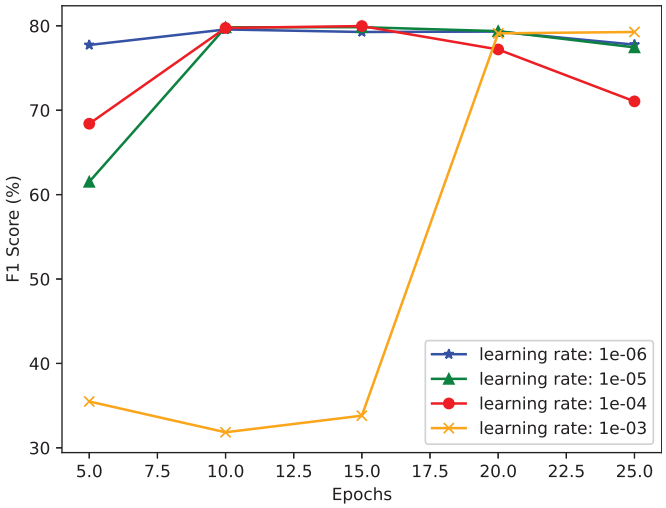

Fig. 5. F1 scores of the MSCRED model at different learning rates using the ADAM optimizer

B. Experimental Setup

1) Data Preprocessing: First, the time series data in the SWaT dataset are downsampled by a factor of ten so as to improve training efficiency. The resulting dataset contains $49,500$ training and $44,992$ test points with $5,463$ anomalies which is about 12.14% of the test set. It is important to note that since the system statuses do not change instantaneously, downsampling by this factor does not have notable negative impact on the model performance. Rather, it offers computational benefits as the required training time is significantly reduced [34].

Then, we normalize each of the time series data using the maximum and minimum values of each time series data to increase the robustness of the algorithm. We perform data normalization to both the training and test sets. The following min-max normalization method is used:

$$\widetilde{x} = -1 + \frac{2 \times (x - min(\mathbf{x}))}{max(\mathbf{x}) - min(\mathbf{x})} \tag{5}$$

where x is the data point, $min(\mathbf{x})$ and $max(\mathbf{x})$ are the minimum and maximum values of each time series data, respectively. The resulting normalized data, $\widetilde{x} \in [-1, 1]$ is used to generate the SSM which are used for model training and testing.

2) Model Design: Since the dataset contains 51 time series, we generate 51 $\times$ 51 SSM to represent each row of data points. We use the following design for MSCRED: 4 convolutional encoders; conv1 to conv4, 32 filters of size 1 $\times$ 1 $\times$ 3, 64 filters of size 1 $\times$ 1 $\times$ 32, 128 filters of size 1 $\times$ 1 $\times$ 64, and 256 filters of size 1 $\times$ 1 $\times$ 128, respectively. Also, we use strides of 1 $\times$ 1 for all the convolutional encoder layers. We choose a step length of 2 for the ConvLSTM. For reconstructing the SSM we use: 4 convolutional decoders; deconv4 to deconv1 with 128 filters of size 1 $\times$ 1 $\times$ 256, 64 filters of size 1 $\times$ 1 $\times$ 128, 32 filters of size 1 $\times$ 1 $\times$ 64, and 3 filters of size 1 $\times$ 1 $\times$ 64, respectively, each with 1 $\times$ 1 stride size [17].

We conduct experiments to determine the best learning rate as shown in Fig. 5. We observe that the largest learning rate, 10^{-3} in our experiments, gives the worst performance in the earlier iterations. At 5 epochs, the learning rate of 10^{-6} gives the best results whereas at 10 and 15 epochs, 10^{-4} performs the best. In order to determine our training scheduler, we conduct experiments with 15 epochs for each training round and a total of 15 epochs for all the training rounds. We observed that conducting experiments with 15 epochs per training round did not improve model performance and significantly prolonged the training time. Hence, we select a learning rate of 10^{-4} and use $E = 15$ epochs as the total number of iterations for each proposed design. For all the proposed curriculum designs that consist of two training rounds, we use 5 epochs in the first and 10 epochs in the final round. For example, in DCL-MSCRED-P2, the first training with the shorter sequence is conducted for 5 epochs and the training with the longer sequence is performed with 10 epochs. For those with three curriculum designs, we select 5 epochs for each training round. We implement the model using PyTorch and train each proposed framework with the ADAM optimizer.

C. Evaluation Metrics

The performance of the implemented frameworks and the SOTA techniques is evaluated on the SWaT dataset using the F1 score which is calculated from precision and recall. Precision evaluates the ratio of data points classified as anomalies that were actually anomalies whereas recall measures the proportion of true anomalies that the model accurately detected. The F1 score is the harmonic mean of precision and recall. These metrics are calculated as:

TABLE I. ANOMALY DETECTION RESULTS USING SWAT DATASET. THE BEST DESIGN AND RESULT IS SHOWN IN BOLD.

Model	Precision (%)	Recall (%)	F1 (%)
DAGMM	27.46	69.52	39.00
AE	72.63	52.63	61.00
LSTM-ED	78.60	66.60	72.10
LSTM-VAE	96.24	59.91	74.01
MSCRED	94.74	69.09	79.91
GAFM	89.10	74.34	81.06
DCL-MSCRED-P2	98.83	69.19	81.40
DCL-MSCRED-P3	97.25	70.01	81.41
DCL-MSCRED-W2	95.10	71.37	81.54
DCL-MSCRED-W3	96.07	71.64	**82.08**
MCLAD	96.25	68.23	79.86
D-MCLAD	94.80	70.82	81.08
Anti-MCLAD	89.59	62.78	73.83

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{8}$$

where TP, FP, and FN represent the true positives, false positives, and false negatives, respectively. We perform the evaluation on the SSM of the test set as shown in Fig. 3. We compare the reconstructed SSM with the input SSM and experimentally determine the threshold which maximizes the F1 score.

D. Anomaly Detection Results

The anomaly detection evaluation results of our proposed approaches and the SOTA frameworks on the SWaT dataset are presented in Table I.

1) Discussion: It can be seen that generally, introducing CL to the model can improve the anomaly detection performance. Overall, DCL-MSCRED-W3 performs the best amongst our designs with an increase of 2.17% in the F1 score compared to MSCRED. The improvement in performance can be attributed to the use of the window size as the difficulty measure. Capturing short local patterns first, facilitates faster convergence and better learning dynamics. This benefits optimization in the subsequent training and by gradually introducing medium and long-term patterns, model generalization is improved which leads to performance improvement. It is also observed that the window size-based CL designs outperform those based on the sequence length. In [35], the researchers use the forecast horizon as the difficulty measure for their forecasting-based framework and observe a significant improvement in performance. Therefore, the window size or forecast horizon is an important feature for CL designs.

The model-based CL design, MCLAD produces the worst results amongst our CL designs. This shows that it does not benefit from CL as the less complex LSTM-ED model is trained with difficult data. As seen by the performance of D-MCLAD, we observe the importance of a CL design for anomaly detection. By introducing data-based CL to MCLAD, an improvement of 1.22% in the F1 score is observed. Therefore, a model-based CL design cannot solely offer notable improvement, however, by training the low capacity model with less complex data, more optimal model parameters can be transferred to the final round of the training to offer better F1 score. Also, to further justify the importance of CL designs, we conduct experiments using an anti-CL technique, hard-to-easy training. In Anti-MCLAD, we first train the more complex MSCRED model with the whole training set and transfer the model parameters to be the initial parameters of the LSTM-ED. We observe a significant drop in performance compared to both MCLAD and MSCRED.

2) Comparison with SOTA Models: We compare our anomaly detection results with related SOTA models using the SWaT dataset. Generally, there is a significant improvement in performance when comparing all our proposed CL-based frameworks to the baseline techniques, MSCRED [17] and LSTM-ED [31].

The autoencoder-based framework (AE) [36] performs better than the DAGMM [21] model due to its ability to learn non-linear correlations in the time series data. The LSTM-based frameworks, LSTM-ED [31] and LSTM-VAE [37] outperform the AE model because of the LSTM network which is capable of adaptively learning the most relevant information and forgetting the less relevant features during training. Furthermore, MSCRED integrates LSTM with CNNs consisting of an attention mechanism to improve the anomaly detection performance. The attention mechanism has a strong capability of learning relevant information required for learning temporal patterns. We also consider GAFM [38], a time series encoding framework which outperforms MSCRED. By introducing CL to the MSCRED framework, all our proposed data-based CL frameworks and D-MCLAD outperform GAFM.

V. CONCLUSION

In this paper, we analyzed the effects of CL and introduced data-based and model-based CL designs for MVTS anomaly detection. First, we examined the importance of sequence length and window size in CL. Then, we investigated whether designing a model-based curriculum would benefit the baseline frameworks. We observed that data-based curriculum designs offer notable performance increases. On the other hand, the model-based curriculum designs did not benefit the baseline MSCRED framework. Hence, we introduced data-based CL to the model-based design and observed improvement in anomaly detection performance. We evaluated the proposed approaches on an open-source anomaly detection dataset, SWaT, and observed an increase in the F1 score in any framework that consists of a data-based curriculum design. In conclusion, CL can improve model performance for deep learning algorithms for MVTS anomaly detection. Our future work will involve analyzing the effect of more data features for CL and extending these approaches to a different case study.

REFERENCES

[1] S. Mokhtari, A. Abbaspour, K. K. Yen, and A. Sargolzaei, "A machine learning approach for anomaly detection in industrial control systems based on measurement data," *Electronics*, vol. 10, no. 4, p. 407, 2021.

[2] M. Evangelou and N. M. Adams, "An anomaly detection framework for cyber-security data," *Computers & Security*, vol. 97, p. 101941, 2020.

[3] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo, "Fraud detection: A systematic literature review of graph-based anomaly detection approaches," *Decision Support Systems*, vol. 133, p. 113303, 2020.

[4] Z. Yang, X. Liu, T. Li, D. Wu, J. Wang, Y. Zhao, and H. Han, "A systematic literature review of methods and datasets for anomaly-based network intrusion detection," *Computers & Security*, vol. 116, p. 102675, 2022.

[5] D. C. Dube, M. Çom, and M. Akar, "Temporal pattern-based collective anomaly detection in textile processes," in *Turkish Automatic Control Conference (TOK)*, vol. 24, 2023, pp. 844–848.

[6] A. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, "Machine learning for anomaly detection: A systematic review," *IEEE Access*, vol. 9, pp. 78 658–78 700, 2021.

[7] P. Kromkowski, S. Li, W. Zhao, B. Abraham, A. Osborne, and D. E. Brown, "Evaluating statistical models for network traffic anomaly detection," in *2019 Systems and Information Engineering Design Symposium (SIEDS)*, 2019, pp. 1–6.

[8] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," *Information Sciences*, vol. 622, pp. 178–210, 2023.

[9] M. Hosseinzadeh, A. M. Rahmani, B. Vo, M. Bidaki, M. Masdari, and M. Zangakani, "Improving security using svm-based anomaly detection: issues and challenges," *Soft Computing*, vol. 25, pp. 3195–3223, 2021.

[10] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, "Curriculum learning: A survey," *International Journal of Computer Vision*, vol. 130, no. 6, pp. 1526–1565, 2022.

[11] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[12] D. C. Dube, Ç. E. Erdem, and Ö. Korçak, "CL-FedFR: Curriculum learning for federated face recognition," in *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 2024, vol. 2, pp. 845–852.

[13] L. Zhang, Z. Mao, B. Xu, Q. Wang, and Y. Zhang, "Review and arrange: Curriculum learning for natural language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3307–3320, 2021.

[14] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 4555–4576, 2021.

[15] D. Wagner, T. Michels, F. C. Schulz, A. Nair, M. Rudolph, and M. Kloft, "Timesead: Benchmarking deep multivariate time-series anomaly detection," *Transactions on Machine Learning Research*, 2023.

[16] 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.

[17] 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, no. 01, 2019, pp. 1409–1416.

[18] J. Chen, D. Pi, and X. Wang, "A two-stage adversarial transformer based approach for multivariate industrial time series anomaly detection," *Applied Intelligence*, pp. 1–20, 2024.

[19] P. Malhotra, L. Vig, G. Shroff, P. Agarwal *et al.*, "Long short term memory networks for anomaly detection in time series." in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2015, pp. 89–94.

[20] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "Deepant: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2018.

[21] 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," in *International conference on learning representations*, 2018.

[22] Z. Zhang, W. Li, W. Ding, L. Zhang, Q. Lu, P. Hu, T. Gui, and S. Lu, "Stad-gan: unsupervised anomaly detection on multivariate time series with self-training generative adversarial networks," *ACM Transactions on Knowledge Discovery from Data*, vol. 17, no. 5, pp. 1–18, 2023.

[23] H. Lim, S. Park, M. Kim, J. Lee, S. Lim, and N. Park, "Madsgm: Multivariate anomaly detection with score-based generative models," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 1411–1420.

[24] D. Hallac, S. Vare, S. Boyd, and J. Leskovec, "Toeplitz inverse covariance-based clustering of multivariate time series data," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 215–223.

[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[26] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," *Advances in neural information processing systems*, vol. 28, 2015.

[27] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. Hauptmann, "Self-paced curriculum learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[28] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 843–852.

[29] L. Brigato and L. Iocchi, "A close look at deep learning with small data," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 2490–2497.

[30] X. Hu, L. Chu, J. Pei, W. Liu, and J. Bian, "Model complexity of deep learning: A survey," *Knowledge and Information Systems*, vol. 63, pp. 2585–2619, 2021.

[31] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[32] A. P. Mathur and N. O. Tippenhauer, "Swat: a water treatment testbed for research and training on ics security," in *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, 2016, pp. 31–36.

[33] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *Critical Information Infrastructures Security: 11th International Conference, CRITIS 2016, Paris, France, October 10–12, 2016, Revised Selected Papers 11*. Springer, 2017, pp. 88–99.

[34] Z. Dong, K. Liu, D. Han, Y. Cao, and Y. Xia, "Reconstruction-based multi-scale anomaly detection for cyber-physical systems," in *2022 4th International Conference on Industrial Artificial Intelligence (IAI)*, 2022, pp. 1–6.

[35] A. Koenecke and A. Gajewar, "Curriculum learning in deep neural networks for financial forecasting," in *Mining Data for Financial Applications: 4th ECML PKDD Workshop, MIDAS 2019, Würzburg, Germany, September 16, 2019, Revised Selected Papers 4*. Springer, 2020, pp. 16–31.

[36] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-based network anomaly detection," in *2018 Wireless Telecommunications Symposium (WTS)*, 2018, pp. 1–5.

[37] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[38] T. Duan, *Unsupervised Multivariate Time Series Anomaly Detection via Transformer-based models and Time Series Encoding*. University of Toronto (Canada), 2021.