International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 6, ISSN No. 2455-2143, Pages 296-305
Published Online October 2022 in IJEAST (http://www.ijeast.com)

ANOMALY DETECTION IN TIME SERIES USING DEEP LEARNING

Md Haris Uddin Sharif, Ketan Gupta, Mehmood Ali Mohammed, Nasmin Jiwani
University of the Cumberlands,
Williamsburg, KY 40769, USA

Abstract— **Time series anomaly detection has ever existed as a fundamental analysis approach. The early time series anomaly detection techniques are mainly statistical and machine learning. For the practical processes of the deep neural network being constantly prospected by experimenters, the result of the deep neural network in anomaly detection tasks has been remarkably more helpful than conventional methods. Conventional models use commanded machine learning algorithms. In the proposed applications, organizing and annotating such a vast** number **of datasets is challenging, time-consuming, or too costly, and it needs specialization learning from professionals in the field. Hence, anomaly detection has become a significant challenge for investigators and practitioners. Anomaly detection is directed as the process of detecting anomaly data instances. In this analysis, we proposed an unsupervised and scalable framework for anomaly detection in time series data. The proposed technique is established on a variational auto-encoder. A deep, productive model that incorporates variational belief with deep learning. Also, real-time analysis has been performed for the time-series data. We used LSTM networks to process, make predictions, and classify based on time series data.**

Keywords— T**ime series, deep learning, neural network, anomaly detection, LSTM, deep neural network, unsupervised learning**

I. INTRODUCTION

Anomaly detection is one of the top research areas for its worldwide nature. In daily life, we notice the abnormalities that highlight our concentration. When something differs from the remains of the allocation, it is labeled as an outlier or anomaly. This paper uses anomalies and outliers interchangeably, as stated in [1]. Anomaly detection directs to finding exact data points that do not work with the specific data set allocation. The most appropriate description of anomaly detection is given by Grubbs [2] for computer science: ''A remote comment, or 'outlier,' is one that deviates markedly from other members of the instance in which it happens''. The word 'anomaly' broadly refers to various issues in various domains. For instance, an anomaly in a network protection method could be an activity correlated to a hacking attempt or malicious software [3].

In the previous years, many techniques have been designed and utilized for anomaly detection in many applications, e.g., network or traffic intrusion detection. They can be classified into three classes: (1) statistical modeling [4-9], (2) information mining-based approaches [10–16], and (3) machine learning-based approaches [17–24]. Many previous studies indicated that they had successfully utilized the above approaches for anomaly classification [5,12,13]; therefore, the computational frameworks highlighting unconventional subsequence detection in time series are yet under-construction. Currently, there is no unified definition of time-series anomalies [25], and they are defined as unusual patterns that do not conform to expected behavior. The anomaly can be defined as a data point wildly dissimilar to other data points [26]. Anomalies can be categorized into three areas [27]: contextual, energy, and collective. Point anomalies are when the data of a single sample are abnormal likened to other data, which can happen in any data. A contextual anomaly defines that the data are unusual in a specific scenario but typical in another system, which can only occur in correlative data. In a time series, time is a context attribute that defines the class of models in the whole sequence. A combined anomaly means that there may be no relationship among numerous individual examples, but they may be abnormal as a group. Each data sample's contextual and behavioral features are considered and define a point anomaly of multivariate time series based on the relationship between a single instance and its historical data. Identification of multidimensional time series inconsistencies is inseparable from multivariate time-series data.
Anomaly detection, the procedure of determining unpredicted objects or events from information, has evolved into a field of attraction for numerous researchers and practitioners and is currently one of the significant studies in data mining and quality assurance [28]. It has been analyzed in various application domains and has experienced significant progress. Classical approaches, adding linear model-based approaches [29], distance-based approaches [30], density-based approaches [31], and support vector machines [32], are always possible options for the algorithm. Moreover, as target approaches become larger and more complex, those methods face limitations, namely an inability to manage multidimensional information or address an insufficiency of labeled anomalies. Moreover, detecting anomalies in time-series data is problematic because it must jointly consider the order and the causality between observations along the time

axis. Currently, multiple methods have been designed to manage these challenges. For example, Hu et al. [33] offered a novel computational technique utilizing a recurrence plot (RP), a square matrix consisting of the times at which a state of a dynamic system recurs. They calculate the local recurrence rates (LREC) by inspecting the RP with a sliding window and detecting anomalies by analogizing similarities between the statistics of the LREC curves.

Deep learning, a subfield of machine learning algorithms inspired by the structure and function of the brain, has been getting attention in recent years. Deep-learning methods learn the complex dynamics in the data while making no assumptions about the underlying patterns. This property makes them the most attractive choice for time-series analysis these days. For example, Yan et al. [34] proposed to combine ensembled long short-term memory (LSTM) neural networks, which memorize long-term patterns in time series, with the stationary wavelet transform (SWT), to forecast the energy consumption. Their experimental outcomes showed that the proposed deep-learning method outperforms classical computational methods.

Due to the growth of industry and the Internet of Things [35], multivariate time-series anomaly detection technology has made significant progress. We can receive faithful time-series data from the devices by configuring a multisensor approach. Moreover, processing these data from sensors is a major problem. First, the data gathered by different sensors may have other particulars, frequencies, and reliance's. However, the pre-processing of these data is very time-consuming and may require some domain knowledge. Jin et al. [36] presented a creative learning framework for multivariate air pollutant attention projection. This approach, which split the particulars and trends by decomposing the actual data into high-frequency and low-frequency parts to learn them in a multi-channel module, delivered a fantastic concept for us to collect the elements of multivariate time series. Moreover, there are some unavoidable problems with the above-said difficulties; for instance, it is challenging to set an exact limit for abnormal and normal data, or the data gathered by other detectors may include noise due to other aspects. Another problem is that the amount of standard data is much more significant than the amount of abnormal information. The issue of highly unbalanced data has become another major problem in time-series anomaly detection [37]. Researchers have tried many ways to process multivariate time-series data.

This analysis goal is to study state-of-the-art deep learning-based anomaly detection methods for time-series data. We know that earlier studies [1], [2], [38]-[43] on this topic do no better than categorizing approaches according to their tools and describing their attributes. In addition to classifying the models according to their methodologies in this article, we further study how they describe interrelationships between variables, learn the temporal context, and identify anomalies in multivariate time series. Therefore, we give instructions to practitioners based on relative practical studies utilizing different standard datasets. Our studies offer practitioners helpful insights for determining the best-suited method(s) for the problem(s) they are trying to solve.

II. LITERATURE REVIEW

In Anomaly detection, along with supervised learning [44], needs a dataset where every example is labeled, and generally, it affects the training of a classifier on the training set. Semi-supervised algorithms like as [45] build a system to present the normal behavior from an input dataset; following the system is utilized to estimate the possibility of the testing dataset to be created by the system. Non-supervised systems like as [46] don't need a labelled dataset and perform under the guessing that the maximum of the data points are standard (e.g., utilizing clustering approaches [47]) and return the remaining ones as outliers.

Yamanishi et al. suggested a Gaussian combination system by calculating every data point and determining the outlier with maximum scores [4]. Zhang and coworkers suggested a mathematical standard to differentiate between normal and abnormal data utilizing statistical algorithms [5]. Kosek et al. created a regression model-based approach for anomaly detection [6]. Goldsein et al. suggested histogram-based outlier detection (HBOS) algorithm, which considers the freedom of the elements, creating it much speedy than multivariate anomaly detection systems. The histogram is needed if the outcomes of outlier detection are ready for use immediately [7]. These systems' limitations are that anomaly detection depends on estimating that the data is generated in a relevant statistical distribution [8].

Resolutions to causing anomaly detection more useful are by utilizing data mining processes, including clustering, or classification. Researchers utilized K-means clustering for similar data points [10, 11] so that the data center found outside of these groups were thought as anomalies. These systems perform in a non-supervised method; moreover, they may not propose correct insights at the demanded class of detail in shorter datasets. Classification-based anomaly detection was also broadly analyzed for real-world applications, e.g., intrusion, traffic, or network detection [12–15]. The objective of classification is to understand from labeled classes of training data for determining classes of new or unknown models [48]. Moreover, good performance demands that the training set must have well-defined labels.

LSTMs have grabbed the concentration of researchers presently in anomaly detection. For example, [49] uses LSTM for forecasting time series and utilizes the forecast mistakes for anomaly detection. They supposed that the resulting forecast errors have a Gaussian distribution, which was utilized then to consider the likelihood of anomalous behavior. Then a point is learned based on the verification dataset to maximize the F-score, which was estimated based on the golden labels within the verification dataset. The method was verified on four-time series. However, [50] follows a same method used to ECG time series, where the forecast errors are appropriate to a Gaussian distribution, and then the threshold is specified based on optimizing the F-score on the verification set, which similarly was estimated based on the provided golden labels. Also, [51] uses an LSTM-based encoder-decoder for multi-sensor anomaly detection. When sufficient anomalous sequences are available, a threshold is learned by maximizing accuracy and memory. The utilize of recurrent neural networks is also typical for intrusion detection, such as in [52], with the aim of identifying and classifying attacks. Moreover, the methods specified above use the golden labels for optimizing the threshold against the prediction errors or building classifiers.

Two primary constraints exist in recent approaches: 1) Most methods, such as statistical and probabilistic methods, are generally appropriate only for univariate datasets where one metric is observed at a time. It can expand to numerous metrics by producing a method for every metric. Moreover, this would not consider any relations between metrics. So, these systems can't smoothly be grown to multivariate research where relations among metrics can be utilized to determine possible anomalous behavior. This is bypassed as Deep AD and can accept as input multiple attributes since it can utilize a single LSTM method that can grab anomalies across numerous attributes, which causes it multivariate. 2) Existing methods normally depend on datasets that include the ground truth labels, where the anomalies are particularly specified to a data point. It is challenging to pick in real-life techniques as labelled data is costly and needs expert knowledge which eventually power be impacted by human errors in labelling the data. Therefore, the quantity of data to be observed and labelled would be unrealistic. Therefore, the primary method might not generalize to new types of anomalies unless retrained and hence needs expert knowledge for the full time of the deployment of the anomaly detection method, causing these techniques unrealistic to be deployed in dynamic environments. The dynamic threshold-based anomaly detection method bypasses this since no labels are required for training or detecting the thresholds [53].

Presently, machine learning approaches are broadly utilized for anomaly detection, involving fuzzy logic [17–19], Bayesian method [20,21], genetic algorithm [18,22], and neural network [23,24]. Nakano et al. offered a fuzzy logic-based anomaly detection approach for network anomaly detection [17]. Hamamto and coworkers generated a mixed strategy for network anomaly detection by utilizing genetic algorithms and fuzzy logic [18]. Mascaro et al. analyzed the usage of Bayesian networks for studying vessel behavior and detecting anomalies [21]. Mixing the dynamic and static networks, they confirmed that their method enhanced the detecting precision in vessel tracks. With the quick growth of artificial intelligence, different neural network methods, e.g., recurrent neural network (RNN) [24] and back propagation neural network (BPNN) [23], were created to observe the anomalies of a difficult method. These methods perform better in some applications; moreover, generalization is difficult.

Compared with standard machine learning approaches, deep learning (DL) has a stronger learning capability and can achieve more precision [54]. The systematic deep learning approaches are malicious productive network (GAN) [55], autoencoder [56], convolutional neural network (CNN) [57], and Long Short-Term Memory (LSTM) [58]. Earlier analyses display that nearly all of the above approaches were applied to anomaly classification [59–61]; moreover, the work highlighting on DL-based abnormal subsequence detection in time series is rarely reported. There are multiple chances to execute anomaly detection in time series utilizing different statistical or SVM-based approaches, with MFAD [9] and LRRDS [33]. Moreover, some endeavors perfectly forecast the unusual sequence in time series utilizing LSTM. However, an excellent deep learning process is needed to execute anomaly detection utilizing LSTM.

There are multiple time-series data, but only a tiny part of the data has been labeled as abnormal or not. It is challenging to develop a clear limit between normal and abnormal data cause they are comparative and connected to the factors in which the information is located. Also, the amount of anomalous data in a sequence is tiny and linked with the desired data. Moreover, multiple researchers have proposed unsupervised techniques for anomaly detection. One of these famous approaches is clustering approaches, e.g., One-Class Support Vector Machine (OCSVM). Ma and Perkins [63] utilized OCSVM to method the training information and determine whether the trial data was abnormal or normal. The mentioned method solves the issue perfectly in many applications. It does not function better on time-series information. The technique only pays attention to the data's knowledge but has nothing to do with capturing the correlation of temporal data. Other system is machine learning approaches, e.g., Isolation Forest (iForest).

Liu et al. [64] offered an approach based on iForest to notice anomalies in sequence data. This method establishes a binary tree by random selection of attributes and cannot receive the connection between different elements, which leads to an inconsistent detection impact. Few researchers considered utilizing predictive approaches to capture temporal data. These predictive methods can forecast what data will occur at the next phase before new data reach. The few usual ideals are the Autoregressive approaches (AR) [65], Moving Average (MA) [66], Autoregressive Moving Average (ARMA) [67], and its variants. These methods perform well in time-series anomaly detection, but they are smoothly impacted by noise. So, there are showing multiple false negative and false positive outcomes when data have serious noise. Nowadays, anomaly detection techniques based on deep learning have made significant improvements. One of the typical deep learning models for processing time series is Long Short-Term Memory (LSTM). An LSTM-based method can forecast the data at the next time step and estimate the distance between

forecasted data and actual time-series data to determine whether it is abnormal. Ergen and Kozat [68] suggested an approach based on the LSTM structure to detect product quality. Lindemann et al. [69] utilized the LSTM system to receive the elements of these fixed-length sequences and confirmed an anomaly detection method based on OCSVM.
Li et al. [70] integrated the features of SAE and LSTM and suggested an anomaly detection approach based on unsupervised deep learning. The LSTM design that can learn the nonlinear relationship of short-term or long-term time-series data is also sensitive to noise, which may expand the risk of misclassification in anomaly detection. Liang et al. [71] estimated the feature matrices and included a forgetting mechanism in the system to reduce the influence of noise. Zhang et al. [72] suggested a system integrating wavelet denoising and primary element analysis to process data noise. The autoencoder approach is another famous noise-friendly approach for detecting sequence data anomalies. It considers whether abnormal by considering the difference between encoded and actual data. Vincent et al. [73] developed a self-encoding noise decrease system that can recover the input data with noise to the data without noise. Borghesi et al. [74] utilized a semi-supervised anomaly detection approach based on an autoencoder. The autoencoder utilizes the concept of encoding and decoding to reduce the noise in time series, making the system more robust. The autoencoder is theoretically better at managing one-dimensional time series [71] because this configuration can't get the relation between multidimensional time-series data. AnoGAN [75] is the initial framework introduced in unsupervised anomaly detection and effectively finds diseased photos from sets of unknown images. Plakias [76] presented GANs based one-class fault detection system for the multi-dimensional issue, and investigations demonstrated the suggested approach outperforms One-class SVM and Isolation Forest. As said, GANs-based architecture has slowly evolved in anomaly detection fields. GAN architecture's main disadvantage is the training fluctuation [62]. If all the above approaches on various techniques impact time-series anomaly detection, they all require indicators to assess anomalies in standard.

III. PROPOSED GOAL

The aim of this project is to detect anomaly in time series data using Deep Learning.

IV. PROBLEM STATEMENT

Anomaly Detection is the identification of the data of a variable or events that do not follow a specific pattern. Anomaly detection helps to identify the data's unexpected behavior with time so businesses and companies can make strategies to overcome the situation. It also helps the firms identify the mistake and scams that will occur at a particular time, or it assists in learning from ex records of data demonstrating typical behavior. We develop LSTM Autoencoder to teach time-series data.

V. DATASET USED

Daily price data for the S&P 500 index has been surprisingly tricky over the years. Most public data only go back to the 2000s. If it is long-term, it is weekly instead of daily. This dataset contains daily closing prices for SPY from 1986 to 2018. The folder contains the CSV file.

The measures follow to detect anomalies in Johnson & Johnson stock price data utilizing an LSTM autoencoder:

- Teach an LSTM autoencoder on Johnson & Johnson's stock value data from 1985–09–04 to 2013–09–03. We suppose that there were no anomalies and that they were predicted.
- Utilizing the LSTM autoencoder to rebuild the mistake on the examination data from 2013–09–04 to 2020–09–03.
- If the rebuild mistake for the examined data is above the point, we mark the data point as an anomaly.

VI. REQUIREMENTS

The main requirements are listed below:

A. Python 3.6.10

Python 3 is a newer version of the Python programming language which was released in December 2008. This version was mainly released to fix problems that exist in Python 2. The nature of these changes is such that Python 3 was incompatible with Python 2. It is backward incompatible. If you see 3.6 or earlier, by the end of December 2021, you'll no longer receive updates or bug fixes. To resolve that, you must upgrade to the latest version of Python. Because 3.6 will have reached End of Life (EOL), it will no longer receive bug fixes, even if they are critical.

B. Keres

Keres is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation. Keres is relatively easy to learn and work with because it provides a python frontend with a high level of abstraction while having the option of multiple back-ends for computation purposes. This makes Keres slower than other deep learning frameworks, but extremely beginner-friendly.

C. Tensorflow

TensorFlow is an open-source framework developed by Google researchers to run machine learning, deep learning and other statistical and predictive analytics workloads. TensorFlow provides a collection of workflows to develop and train models using Python or JavaScript, and to easily deploy in the cloud, on-prem, in the browser, or on-device no matter what language you use. The tf. data API enables you to build complex input pipelines from simple, reusable pieces.

D. Numpy

NumPy, meaning Numerical Python, is a library containing multi-dimensional array entities and a group of exercises for processing those arrays. Utilizing NumPy, mathematical and logical functions on shows can be executed. It also has procedures for working in the domain of linear algebra, Fourier transform, and matrices. NumPy was developed in 2005 by Travis Oliphant. It is an open-source object, and you can utilize it freely.

E. Pillow 6.1

Pillow 6.1 is a Python Imaging Library. It is a free and open-source library for the Python programming language that includes support services for saving many different image file formats, opening, and manipulating. It is open for Linux, Windows, and Mac OS X. The most recent version of PIL is 1.1.

F. Scikit-learn

Scikit-learn is an integral part of the Python machine learning toolkit at JPMorgan. It is broadly utilized across all aspects of the bank for Classification, estimate analytics, and many other machine learning jobs. Scikit-learn is an open-source data analysis library and the gold standard for Machine Learning (ML) in the Python ecosystem. The main visions and elements contain Algorithmic decision-making methods, including Classification: recognizing and classifying data based on designs.

G. Pandas

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data. The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008. Pandas is an open-source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.

H. Seaborn

Seaborn is a Python data library established on matplotlib. It delivers a high-level interface for creating gorgeous and informative statistical graphics. You can study the preliminary notes or the article for a quick preface to the concepts behind the library.

I. Matplotlib

Matplotlib is the Python programming language for plotting libraries for the numeric mathematical extension NumPy. It uses general-purpose GUI toolkits such as wxPython, Tkinter, GTK, or Qt. A systematic "Pylab" interface is based on a state-of-the-art machine (such as OpenGL), providing an object-based API for embedding stories in applications, which is similar to MATLAB, though its usage is disheartened closely. SciPy creates the use of Matplotlib.

J. Tqdm

The name tqdm comes from the Arabic name takaddum, which means 'progress.' The Spanish acronym for "I love you so much" (te quiero demasiado). Applying tqdm can be done effortlessly on our loops, functions, or even pandas.

VII. EXPERIMENT

Training:

- Data distribution:

80% Training & validation and 20% Test.

- Loss Function:

Mean Absolute Error.

- Evaluation Metrics:

Mean Absolute Error

- Optimizer:

Adam Optimizer.

VIII. MODEL

LSTM Autoencoder

LSTM Networks:

The LSTM network is a revised version of RNNs, efficiently remembering the long-term allegiance in data. While this issue solves in LSTM networks, RNNs confront the vanishing gradient problem. The memory unit (or cell) is the principal of the LSTM network, as shown in Figure 1. A cell is made up of three sigmoid and a single tanh layer to organize information outside and inside a cell. The forget gate can reset the memory unit with a sigmoid function. To manage the output and input data flow in the memory unit, output and input gates are provided, respectively.

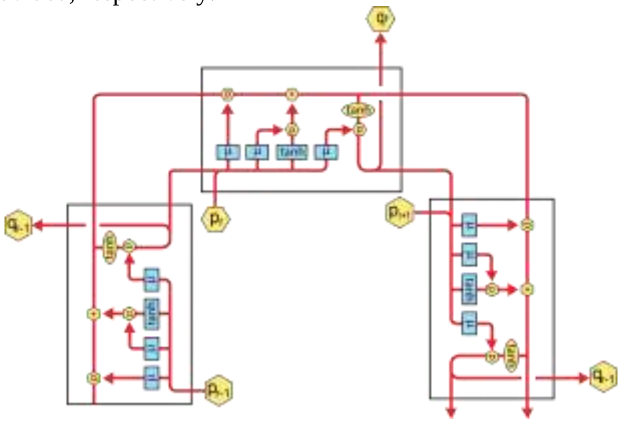

Fig. 1. The Architecture of an LSTM cell.

A. Autoencoder

Fig. 1. An ANN that has two distinct parts is an autoencoder. The encoder h = f (x) converts the original data, and the decoder reconstructs the data, which formulates as xˆ = g (h). Creating autoencoders aims to replicate the network's input to its output. The system is pushed to prioritize the

valuable input factors; on the other hand, an autoencoder plays a vital role in learning the valuable features of data. Thus, the networks don't understand to copy correctly but approximately, and they are limited in such a manner that develops output that resembles the training information.

B. Autoencoder LSTM

In this suggested method, an autoencoder compresses and encodes the data. Autoencoder is an unsupervised ANN that creates a more miniature encoded expression of data and then knows how to rebuild the data. The resemblance of the rebuilt data to the initial input defines the efficacy of an autoencoder. The predicted workflow utilized in this article is shown in Fig. 3. The LSTM level consists of autoencoders with encoders and decoders. After every LSTM layer, a put dropout layer to protect the method from overfitting in the training process. First, the autoencoder is taught. Next, the encoder part is utilized as the element generator. The last phase is to train the LSTM-based predictor to deliver the modified closing price projection for the next day.

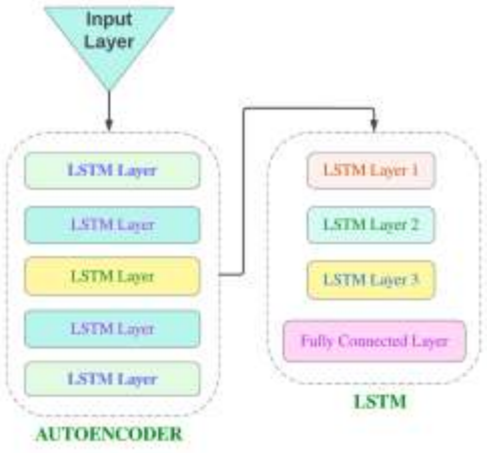

Fig. 2. The architecture of Autoencoder LSTM.

We describe the rebuild LSTM Autoencoder architecture that predicts input lines with 30-time phases and outputs a string with 30-time actions and one element. Repeat Vector () repeats the inputs 30 times. Set return sequences=True, so the output will still be a sequence. Time Distributed (Dense (X_train.shape [2])) is added at the end to get the output, where X_train.shape [2] is the number of features in the input data. The whole parameters are 198,273. All parameters are trainable, and a dropout layer is utilized to bypass the overfitting issue.

Table -1 Experiment Result

Model: "sequential"

Layer (type)	Output shape	Param #
lstm (LSTM)	**(None, 128)**	**66560**
dropout (Dropout)	**(None, 128)**	**0**
repeat_vector (RepeatVector)	**(None, 30, 128)**	**0**
lstm_1 (LSTM)	**(None, 30, 128)**	**131584**
dropout_1 (Dropout)	**(None, 30, 128)**	**0**
time_distributed (TimeDistributed)	**(None, 30, 1)**	**129**

Total params: 198,273
Trainable params: 198,273
Non-trainable params: 0

C. Training and Validation Loss

On the entire group of data, the LSTM closed teaching at epoch 10. The project losses in the last epoch came at the price of 0.0736, while the confirmation loss came at the value of 0.0455. The training and verification losses for each period are presented in Fig. 3.

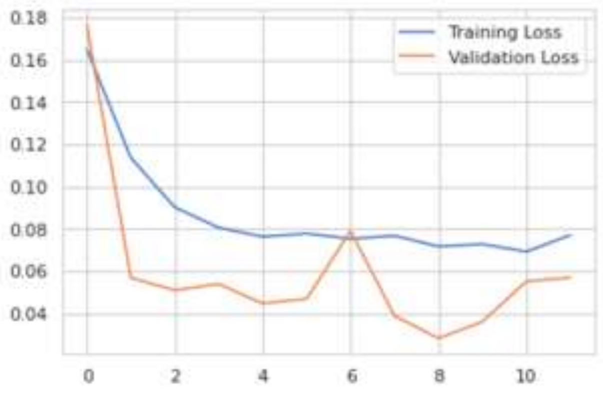

Fig. 3. Training and Validation Loss.

D. Train and Test Loss Histogram

Histogram prices of teaching data (members) and examine data (non-members). We see the more significant divergence between the loss allocation over members and non-members in the powerful approach equivalent to the natural system.

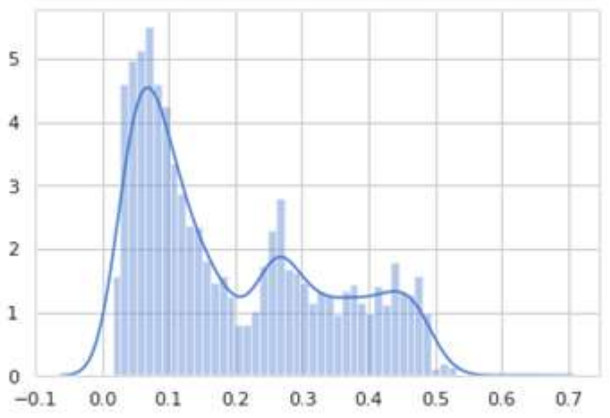

Fig. 4. Train and Test Loss Histogram.

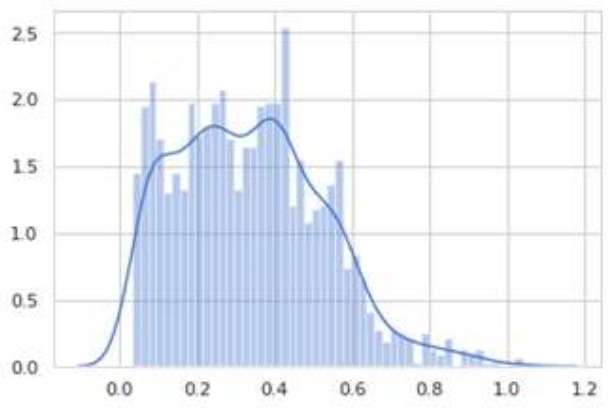

Fig. 5. Train and Test Loss Histogram.

E. Anomaly Detected

We will find anomalies by specifying how well the system rebuilds the input data.

- Find MAE losing on teaching examples.
- Discover most MAE loss prices. This is the most harmful our method has executed attempting to rebuild an example. We will create this threshold for anomaly detection.
- If the rebuild loss, for example, is better than this threshold price, then we can assume that the method sees a pattern it isn't acquainted with. We will mark this example as an anomaly.

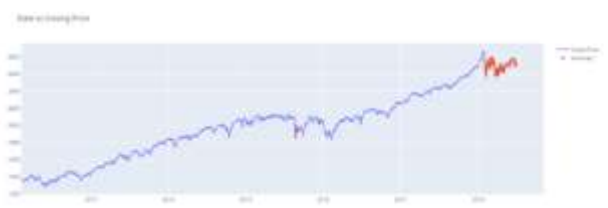
Fig. 5. Train and Test Loss Histogram

VII.CONCLUSION

A total of 27 slender beams without transverse reinforce- ment and 18 slender beams with transverse reinforcement were tested experimentally to study the behaviour for dif- ferent grades i.e. M1, M2 and M3 mixes.

VIII. REFERENCE

[1] Assendorp J. P. (2017). Deep learning for anomaly detection in multivariate time series data, Faculty of Engineering and Computer Science, Department of Computer Science, Hochschule für angewandte Wissenschaften Hamburg, Hamburg, Germany. (pp. 1-90)

[2] Blázquez-García A., Conde A., Mori U., and Lozano J. A. (2020). A review on outlier/anomaly detection in time series data, Cornel University, https://doi.org/10.48550/arXiv.2002.04236

[3] Shyu M.-L., Chen S.-C., Sarinnapakorn K., and Chang, L. (2003). A novel anomaly detection scheme based on principal component classifier. IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM'03), https://apps.dtic.mil/sti/pdfs/ADA465712.pdf

[4] Yamanishi K., Takeuchi J.-i., Williams G., and Milne P. (2004). On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms, Data Mining and Knowledge Discovery 8(3), 275–300.

[5] Zhang J., and Zulkernine M. (2006). Anomaly based network intrusion detection with unsupervised outlier detection, in Proceedings of the IEEE International Conference Communication, pp. 2388–2393.

[6] Kosek A. M., and Gehrke O. (2016). Ensemble regression model-based anomaly detection for cyber-physical intrusion detection in smart grids, in Proceedings of the Ieee Electronic Power Energy Conference, Urbana,IL, USA. https://doi.org/10.1109/EPEC.2016.7771704

[7] Goldstein M., and Uchida S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, https://doi.org/10.1371%2Fjournal.pone.0152173

[8] Veasey T. J., and Dodson S. J. (2014). Anomaly detection in application performance monitoring data, International Journal of Machine Learning and Computing 4(2), 120–126.

[9] Hu M., Ji Z., Yan K., Guo Y., Feng X., Gong J., Zhao X., Dong L. (2018). Detecting anomalies in time series data via a meta-feature based approach, IEEE Access 6, 27760–27776. https://doi.org/10.1109/ACCESS.2018.2840086

[10] Ravale U., Marathe N., and Padiya P. (2015), Feature selection-based hybrid anomaly intrusion detection

system using K means and RBF kernel function, Procedia Computer Science45, 428–435.

[11] Muniyandi A. P., Rajeswari R., and Rajaram R. (2012). Network anomaly detection by cascading K-means clustering and C4.5 decision tree algorithm, Procedia Engineering 30, 174–182.

[12] Mamun S. M. A. A., and Valimaki J. (2018). Anomaly detection and classification in cellular networks using automatic labeling technique for applying supervised learning, Procedia Computer Science, 140, 186–195.

[13] Barria J. A., and Thajchayapong S. (2011). Detection and classification of traffic anomalies using microscopic traffic variables, IEEE Transactions on Intelligent Transportation Systems 12(3), 695–704.

[14] Iglesias F., and Zseby T. (2015). Analysis of network traffic features for anomaly detection, Machine Learning 101(1-3), 59–84.

[15] Tao X. L., Peng Y., Zhao F., Zhao P. C., and Wang Y. (2018). A parallel algorithm for network traffic anomaly detection based on isolation forest, International Journal of Distributed Sensor Networks 14(11).

[16] HuX M., Feng, Ji Z., Yan K., and Zhou S. (2019). A novel computational approach for discord search with local recurrence rates in multivariate time series, Information Sciences 477, 220–233.

[17] Nakano M., Takahashi A., and Takahashi S. (2017). Fuzzy logic-based portfolio selection with particle filtering and anomaly detection, Knowledge-Based Systems 131, 113–124.

[18] Hamamoto A. H., Carvalho L. F., Sampaio L. D. H., Abrão T., and Proença M. L. (2018). Network anomaly detection system using genetic algorithm and fuzzy logic, Expert Systems with Applications 92, 390–402.

[19] Li Q., Sun R., Wu H., and Zhang Q. (2018). Parallel distributed computing based wireless sensor network anomaly data detection in IoT framework, Cognitive Systems Research 52, 342–350.

[20] Akouemo H. N., and Povinelli R. J. (2016). Probabilistic anomaly detection in natural gas time series data, International Journal of Forecasting 32(3), 948–956.

[21] Mascaro S., Nicholso A. E., and Korb K. B. (2014). Anomaly detection in vessel tracks using Bayesian networks, International Journal of Approximate Reasoning 55, (1), 84–98.

[22] Li Y., Guo L., Tian Z.-H., and Lu T.-B. (2008). A lightweight web server anomaly detection method based on transductive scheme and genetic algorithms, Computer Communications 31(17), 4018–4025.

[23] Chiba Z., Abghour N., Moussaid K., Omri A. E., and Rida M. (2018). A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection, Computers & Security 75, 36–58.

[24] Kubota T., and Yamamoto W., (2019). Anomaly detection from online monitoring of system operations using recurrent neural network, Procedia Manufacturing 30, 83–89.

[25] Canizo M., Triguero I., Conde A., and Onieva E. (2019). Multi-head CNNRNN for multi-time series anomaly detection: An industrial case study, Neurocomputing 363, 246–260.

[26] Wang H., Bah M. J., and Hammad M. (2019). Progress in outlier detection techniques: A survey, IEEE Access 7, 107964–108000.

[27] Chandola V., Banerjee A., and Kumar V. (2009). Anomaly detection: A survey, ACM Comput. Surv. 41(3), 15. DOI: 10.1145/1541880.1541882.

[28] Blázquez-García A., Conde A., Mori U., and Lozano J. A. (2020). A review on outlier/anomaly detection in time series data, Cornel University. Available: https://doi.org/10.48550/arXiv.2002.04236

[29] Shyu M.-L., Chen S.-C., Sarinnapakorn K., and Chang L. (2003). A novel anomaly detection scheme based on principal component classifier, Dept. Elect. Comput. Eng., Univ. Miami, Coral Gables, FL, USA, Tech. Rep.

[30] Angiulli F., and Pizzuti C. (2002). Fast outlier detection in high dimensional spaces, in Proc. Eur. Conf. Princ. Data Mining Knowl. Discovery (PKDD). Berlin, Germany: Springer, 15–27.

[31] Breunig M. M., Kriegel H.-P., Ng R. T., and Sander, J. (2000). LOF: Identifying density-based local outliers, in Proc. ACM SIGMOD Int. Conf. Manage. Data, 93–104.

[32] Schölkopf B., Platt J. C., Shawe-Taylor J. C., Smola A. J., and Williamson R. C. (2001). Estimating the support of a high-dimensional distribution, Neural Comput. 13(7), 1443–1471.

[33] Hu M., Feng X., Ji Z., Yan K., and Zhou S. (2019). A novel computational approach for discord search with local recurrence rates in multivariate time series, Inf. Sci., 477, 220–233.

[34] Yan K., Li W., Ji Z., Qi M., and Du Y. (2019). A hybrid LSTM neural network for energy consumption forecasting of individual households, IEEE Access 7, 157633–157642.

[35] Yaqoob I., Ahmed E., Hashem I. A. T., Ahmed A. I. A., Gani A., Imran M., and Guizani M. (2017). Internet of Things architecture: Recent advances, taxonomy, requirements, and open challenges, IEEE Wireless Commun. 24 (3), 10–16. DOI: 10.1109/MWC.2017.1600421.

[36] Jin N., Zeng Y., Yan K., and Ji Z. (2021). Multivariate air quality forecasting with nested LSTM neural network, IEEE Trans. Ind. Informat., early access. DOI: 10.1109/TII.2021.3065425.

[37] Fernández A., Garcia S., Galar M., Prati C. R., Krawczyk B., and Herrera F. (2018). Learning from

Imbalanced Data Sets, Cham, Switzerland: Springer, 1–377.

[38] Chalapathy R., and Chawla S. (2019). Deep learning for anomaly detection: A survey, Cornel University. Available: http://arxiv.org/abs/1901.03407

[39] Fawaz H. I., Forestier G., Weber J., Idoumghar L., and Müller P.-A. (2019). Deep learning for time series classification: A review, Data Mining Knowl. Discovery, 33(4), 917–963.

[40] Cook A. A., Mısırlı G., and Fan Z. (2020). Anomaly detection for IoT time-series data: A survey, IEEE Internet Things J., 7(7), 6481–6494.

[41] Talagala P. D., Hyndman R. J., Smith-Miles K., Kandanaarachchi S., and Muñoz M. A., (2020). Anomaly detection in streaming nonstationary temporal data, J. Comput. Graph. Statist., 29(1), 13–27.

[42] Goldstein M., and Uchida S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, PLoS ONE, 11(4), Art. no. e0152173.

[43] Aggarwal C. C. (2015). Outlier analysis, in Data Mining. New York, NY, USA: Springer, 237–263.

[44] Goernitz N., Kloft M., Rieck K., and Brefeld U. (2013). Toward Supervised Anomaly Detection, Journal of Artificial Intelligence Research.

[45] Noto K., Brodley C., and Slonim D. (2012). FRaC: a feature-modeling approach for semi-supervised and unsupervised anomaly detection, Data Min Knowl Disc, 25(1), 109–133. https://doi.org/10.1007/s10618-011-0234-x

[46] Amer M., Goldstein M., and Abdennadher S. (2013). Enhancing one-class support vector machines for unsupervised anomaly detection, In: ACM SIGKDD. 8–15.

[47] Rajasegarar S., Leckie C., Palaniswami M. (2014). Hyperspherical cluster based distributed anomaly detection in wireless sensor networks, Journal of Parallel and Distributed Computing, 74(1), 1833–1847.

[48] Jain A., Verma B., and Rana J. L. (2014). Anomaly intrusion detection techniques: a brief review, International Journal of Scientific & Engineering Research, 5(7), 1372–1383.

[49] Malhotra P., Vig L., Shroff G., Agarwal P. (2015). Long short-term memory networks for anomaly detection in time series, In: ESANN.

[50] Chauhan S., and Vig L. (2015). nomaly detection in ecg time signals via deep long shortterm memory networks, In: IEEE DSAA, 1–7.

[51] Malhotra P., Ramakrishnan A., Anand G., Vig L. (2015). Agarwal, P., Shroff, G., “Lstm-based encoder-decoder for multi-sensor anomaly detection, https://doi.org/10.48550/arXiv.1607.00148.

[52] Al-Jarrah O., and Arafat A. (2015). Network intrusion detection system using neural network classification of attack behavior, Journal of Advances in Information Technology, 6(1).

[53] Buda T. S., Caglayan B., and Assem, H. (2018). Deep AD: A Generic Framework based on Deep Learning for Time Series Anomaly Detection, in book: Advances in Knowledge Discovery and Data Mining, 577-588. http://dx.doi.org/10.1007/978-3-319-93034-3_46

[54] Feng F., Liu X., Yong B., Zhou R., and Zhou Q. (2019). Anomaly detection in ad-hoc networks based on deep learning model: a plug and play device, Ad Hoc Networks, 84, 82–89.

[55] Lin Y., Li L., Jing H., Ran B., and Sun D. (2020). Automated traffic incident detection with a smaller dataset based on generative adversarial networks, Accident Analysis & Prevention, 144, Article ID 105628.

[56] Xie W., Lei J., Liu B., Li Y., and Jia X. (2019). Spectral constraint adversarial autoencoders approach to feature representation in hyperspectral anomaly detection” Neural Networks, 119, 222–234.

[57] N. Chouhan, A. Khan, and H.-u.-R. Khan, (2019) “Network anomaly detection using channel boosted and residual learning based deep convolutional neural network” Applied Soft Computing, 83, Article ID 105612.

[58] Kim T.-Y., and Cho S.-B. (2018). Web traffic anomaly detection using C-LSTM neural networks, Expert Systems with Applications, 106, 66–76.

[59] Erfani S. M., Rajasegarar S., Karunasekera S., and Leckie C. (2016). High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning, Pattern Recognition, 58, 121–134.

[60] Li L., Du B., Wang Y., Qin L., and Tan H. (2020). Estimation of missing values in heterogeneous traffic data: application of multimodal deep learning model, Knowledge-Based Systems, 194, Article ID 105592.

[61] Li L., Lin Y., Du B., Yang F., and Ran B. (2020). Real-time traffic incident detection based on a hybrid deep learning model, Transportmetrica A: Transport Science, 1813214.

[62] Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A. C., and Bengio Y. (2014). Generative adversarial nets, in Proc. Adv. Neural Inf. Process. Syst., 2672–2680.

[63] Ma J., and Perkins S. (2003). Time-series novelty detection using one-class support vector machines, in Proc. IJCNN, 3. IEEE, 1741–1745.

[64] Liu T. F., Ting M. K., and Zhou Z.-H. (2012). Isolation-based anomaly detection, ACM Trans. Knowl. Discovery Data, 6 (1), 1–39.

[65] Xiao-Chao D., and Lin Y. (2021). Traffic flow prediction based on multivariate linear AR model, Comput. Eng., 38(1), 90–92, and 95.

[66] Patel R., and Saha G. (2018). Time series regression of weather parameters over the last century for cotton

crop, Proc. Int. Conf. Intell. Syst. Signal Process., 125–153.

[67] Wang Y., Huang K., and Tan T. (2007). Group activity recognition based on ARMA shape sequence modeling, in Proc. ICIP, 209.

[68] Ergen T., and Kozat S. S. (2020). Unsupervised anomaly detection with LSTM neural networks, IEEE Trans. Neural Netw. Learn. Syst., 31, (8), 3127–3141.

[69] Lindemann B., Jazdi N., and Weyrich M. (2019). Detektion von anomalien zur Qualitätssicherung basierend AUF sequence-to-sequence LSTM netzen, at-Automatisierungstechnik, 67(12), 1058–1068. DOI: 10.1515/auto-2019-0076.

[70] Li Z., Li J., Wang Y., and Wang K. (2019). A deep learning approach for anomaly detection based on SAE and LSTM in mechanical equipment, Int. J. Adv. Manuf. Technol., 103(1–4), 499–510.

[71] Liang H., Song L., Wang J., Guo L., Li X., and Liang J. (2021). Robust unsupervised anomaly detection via multi-time scale DCGANs with forgetting mechanism for industrial multivariate time series, Neurocomputing, 423, 444–462.

[72] Zhang S., Tang Q., Lin Y., and Tang Y. (2017). Fault detection of feed water treatment process using PCA-WD with parameter optimization, ISA Trans., 68, 313–326.

[73] Vincent P., Larochelle H., Bengio Y., and Manzagol P.-A. (2008). Extracting and composing robust features with denoising autoencoders, in Proc. 25th Int. Conf. Mach. Learn. (ICML), 1096–1103.

[74] Borghesi A., Bartolini A., Lombardi M., Milano M., and Benini L. (2019). A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems, Eng. Appl. Artif. Intell., 85, 634–644.

[75] Schlegl T., Seeböck P., Waldstein S. M., Schmidt-Erfurth U., and Langs G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery, in Information Processing in Medical Imaging (Lecture Notes in Computer Science), 10265, 146–157.

[76] Plakias S., and Boutalis Y. S. (2019). Exploiting the generative adversarial framework for one-class multi-dimensional fault detection, Neurocomputing, 332, 396–405, https://doi.org/10.1016/j.neucom.2018.12.041