Temporal graph convolutional autoencoder based fault detection for renewable energy applications.

ARIFEEN, M. and PETROVSKI, A.

2024

Temporal Graph Convolutional Autoencoder based Fault Detection for Renewable Energy Applications

1st Murshedul Arifeen
School of Computing
Robert Gordon University
Aberdeen, Scotland
d.arifeen@rgu.ac.uk

2nd Andrei Petrovski
National Subsea Centre
Robert Gordon University
Aberdeen, Scotland
a.petrovski@rgu.ac.uk

***Abstract*—Detecting faults in energy generation systems is a challenging task due to the complex nature of the system, measurement noise, and outliers. Recently, researchers have shown an increasing interest in using data-driven models that utilize sensor data for fault detection and diagnosis. However, the nonlinearities, spatial and temporal dependencies in time-series sensor data make it difficult to develop an effective data-driven fault detection model. To address this issue, we propose an autoencoder model that uses a temporal graph convolutional layer to detect faults in the energy generation process. The proposed model has exceptional spatiotemporal feature learning capabilities, making it ideal for fault detection applications. In addition, we have included a data processing module to reduce noise and eliminate outliers from sensor data. We evaluated the model's performance using wind turbine blades and photovoltaic microgrid datasets. Experimental results have demonstrated that the proposed model outperforms other fault detection models based on graph convolutional autoencoders.**

***Index Terms*—Temporal Graph Convolutional Autoencoder, Fault Detection, Wind Turbine, Photovoltaic Microgrid**

I. Introduction

Fossil fuels have been the primary source of energy for decades, but their combustion emits greenhouse gases, causing global warming [1]. Renewable energy is a clean, abundant, and sustainable alternative, including solar, wind, tidal, and geothermal. Although reliable, technical faults can damage the energy generation process, making it critical to identify and diagnose the root cause of these faults [1] [2].

There are two methods for detecting faults in a system: model-driven and data-driven approaches [3] [4]. The former creates a physical model, while the latter relies on Supervisory Control and Data Acquisition (SCADA) sensor data [4]. These data-driven models can be implemented using shallow or deep learning (DL) techniques, making them reliable and efficient solutions for fault detection over model-driven approaches. DL is more effective than shallow learning in analyzing complex SCADA data, and there are two approaches to fault detection using DL: forecasting and reconstruction-based techniques [2]. This study focuses solely on reconstruction-based techniques.

Autoencoders (AE) (such as long short-term memory (LSTM) or convolutional neural network (CNN)-based stacked AE, variational, sparse, or denoising AE) are reconstruction-based models used in industrial applications for fault detection [5]. However, most models have limitations in learning spatial and temporal relations from SCADA data [6] [7]. It is necessary to consider spatial and temporal relations simultaneously to extract comprehensive information from SCADA data [7]. Graph neural networks (GNN) can learn spatiotemporal data effectively due to their unique features, such as permutation invariance, local connectivity, and computationality [2]. Researchers have utilized GNN, particularly graph convolutional networks (GCN), to construct spatiotemporal AE for identifying and diagnosing faults in various process monitoring applications, outperforming traditional AE models [6] [8] [9] [10] [11]. The recently proposed model, temporal graph convolutional network (T-GCN) combining GCN and gated recurrent unit (GRU), can exceed the GCN in learning spatiotemporal SCADA data by explicitly learning the temporal pattern through the GRU [12].

This paper uses the T-GCN model to build an AE for detecting faults in renewable energy applications such as wind turbines and photovoltaic microgrids. The T-GCN's ability to learn spatiotemporal features makes it an ideal choice for complex nonlinear spatiotemporal modelling of SCADA data. The GCN component of TGCN learns the spatial dependence of SCADA features, while the GRU learns temporal dependencies [12]. We have also used skip connections between the layers to reduce information loss during the encoding-decoding process. The main contributions of this paper are summarized as follows.

- A T-GCN layer-based AE model is developed for detecting faults in energy industry applications. The T-GCN model can effectively learn the spatiotemporal nature of the SCADA data compared to the GCN-based AE model.
- We used the Fast Fourier Transform (FFT) to reduce noise, boxplot, and isolation forest (IF) to remove outliers from the highly noisy SCADA data.
- We compared the performance of the TGCN-AE model with GCN AE using wind turbine blades and photovoltaic microgrid datasets.
- T-GCN AE outperforms GCN-AE in false alarm rate (FAR) and fault detection rate (FDR).

Section II briefly discussed the recently published GCN-based forecasting or reconstruction models for fault detection works. Section III has addressed the noise reduction technique,

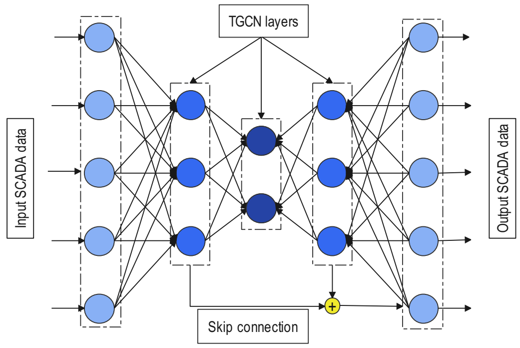

Fig. 1: Architecture of an Autoencoder where the layers are replaced by the T-GCN layer. Also, skip connections are added to reduce the information loss during the feature compression and reconstruction process.

outlier removing methods, and T-GCN AE-based fault detection model. Then, in section IV, we explained the dataset, experimental procedure, and comparison results of T-GCN regarding FAR and FDR. Finally, section V concludes this paper.

II. Literature Review

Data-driven models such as neural networks, CNN, and LSTM-based AE are highly effective in detecting faults by monitoring the deviation of real-time data from the learned patterns. Still, they need to improve in learning spatiotemporal data simultaneously. For instance, CNN-based models recognize only local spatial features [6] [12]. In contrast, the authors in [6] proposed a GCN AE-based fault detection model that uses graph structure data to understand spatiotemporal relations from data, thus detecting faults more accurately. Liu et al. [8] introduce a novel graph dynamic autoencoder method that uses GCN to monitor the Tennessee Eastman process. Yu et al. [3] proposed a fast deep graph convolutional network to diagnose wind turbine gearbox faults using wavelet decomposition, resulting in a high fault recognition accuracy of 99.60% Similarly, Yang et al. [13] proposed a 94.75% accurate GNN and one-shot learning-based model for wind turbine gearbox monitoring. Lai et al. [14] presented a wavelet-driven GCN for detecting blade icing in wind turbines by capturing multiscale features of SCADA data in time and frequency domains. A GCN-SA hybrid model is proposed for fault diagnosis in a traction system by combining the GCN model and prior system knowledge [4]. Liu et al. [7] proposed a novel spatiotemporal model that monitors wind turbine conditions by accurately learning multiple features' spatial and temporal dependencies, combining a GCN and GRU. Multiple spatiotemporal blocks are stacked to extract high-level features from the graph-structured sensor data. GNN-based model is also proposed in [2] to learn complex interactions and coupling relations between sensors of wind turbines for early fault diagnosis. The review shows that the GCN-based model is becoming popular for condition monitoring by learning spatiotemporal features. It uses forecasting to identify faults, while GCN AE-based models use a reconstruction approach. However, T-GCN based AE model is not well studied for fault detection problems.

III. System Model

A. Noise Reduction using FFT

Reducing noise is vital in enhancing DL models, and one way to do that is by using the Fourier Transform (FT) to transform time domain data into the frequency domain. FT has proven to be a highly efficient tool for reducing noise in various applications [15] [16]. In this study, we have used the FFT algorithm to eliminate noisy components from the SCADA data by decomposing a complex function (signal) into a linear combination of trigonometric functions (sinusoidal signals) [17]. Mathematically, we can represent the FT of a function $f(t)$ as: $\tilde{f}(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-i\omega t}dt$, where, $\tilde{f}$ is the FT of f which depends on the frequency ω. However, the SCADA data from wind turbines and microgrid are discrete in practice. Therefore, we apply discrete FT (DFT) as: $Z_k = \sum_{j=0}^{N-1} X_j e^{\frac{-2\pi i}{N}kj}$, where, Z_k is the $k'th$ element of the DFT and X_j is the original SCADA data. The FFT algorithm produces the one dimensional DFT of the input data, and multiplying the output with the conjugate produces the noisy input SCADA data's power spectral density (PSD). From the PSD, the dominant components of the time series data can be identified, and other (noisy) components can be eliminated by thresholding.

B. Outlier Removal

We used Boxplot and IF to remove SCADA data outliers. One outlier removal method wasn't enough, so we first used Boxplot and then IF to suppress the remaining outliers.

1) Boxplot: Boxplot is a simple and effective way to eliminate outliers from the data [18]. Boxplot generates a five-number summary of a data series [19]: the minimum value (Min), the maximum value (Max), the first quartile $(Q1)$, the median value (the second quartile $(Q2)$), the third quartile $(Q3)$. This method produces the $(Max - Min)$ range and the interquartile range$(IQR(Q3 - Q1))$; using these ranges, we can get a boundary to distinguish the outliers from the standard data samples. Mathematically, any data points above $(Q3 + 1.5 \times IQR)$ and below $(Q1 - 1.5 \times IQR)$ are outliers, and we can remove these data points from the original data series.

2) Isolation Forest: The IF algorithm is a variation of the Decision tree algorithm [20]. It identifies outliers by randomly selecting a feature from the set of features and then selecting a split value between that feature's maximum and minimum values. This random partitioning of features creates shorter paths in trees for anomalous data points, making them distinguishable from the rest of the data.

C. Autoencoder

AE is a robust DL algorithm that excels in learning data representation. It has an encoder, a decoder, and a latent representation layer [21]. AE uses unsupervised learning and is versatile for various tasks. Let's consider an unlabelled training

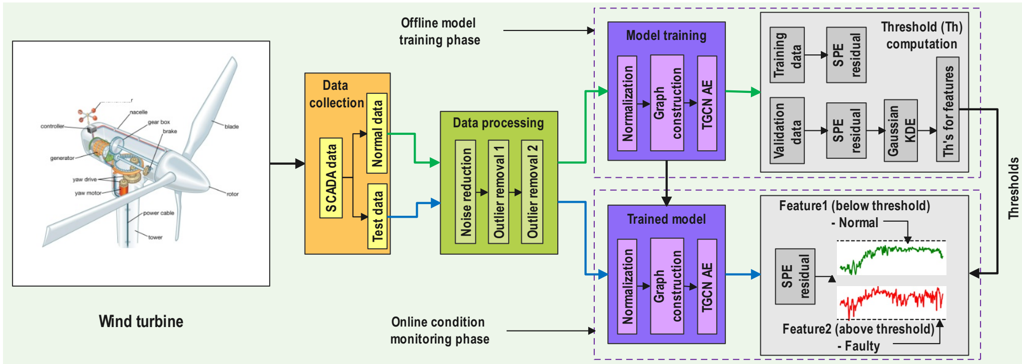

Fig. 2: TGCN AE model based faulty condition detection process. The normal data collected from the SCADA system is first used to train the TGCN AE model after cleaning (noise reduction and outlier elimination) through the data processing module.

dataset X consisting of N samples x_i with dimension n. Then the encoder function can be defined as $\beta_i = g(x_i)$, where β_i is the latent representation layer with dimension q. The encoder reduces the input data dimension from n to q. The decoder reconstructs the input data from β_i of dimension q back to n. An AE trains by minimizing a loss function that represents the difference of reconstructed samples $\bar{x}$ and the original samples x using a learning algorithm. The typical loss function for a deterministic AE is Mean Square Error (MSE) [22], i.e., $Loss = \frac{1}{N}\sum_i^N |x_i - \bar{x}_i|^2$.

D. Temporal Graph Convolutional Network

GCN based DL models work on graph structured data. A graph can be denoted as $G = (V, E, A)$, which comprises a set of nodes V, $|V| = K$, a set of edges E, $|E| = L$, and an adjacency matrix A. The adjacency matrix $A \in R^{K\times K}$ represents the weights and edges among the nodes V. That is, if there is an edge between node $v_i \in V$ and $v_j \in V$, then they are neighbors $(i \neq j)$, and the entry $A(i, j)$ in the adjacency matrix A denotes the weights of their edge. The weights of the edges can be computed through various techniques, for example, euclidean similarity, correlation matrix, or cosine similarity. On the contrary, for an unweighted graph, the entries of the adjacency matrix A can be set as $(i, j) = 1$. Time series data needs to be transformed into graph structure data to use a GCN based model. We followed the procedure of GDA [8] to construct graph attributes from SCADA data.

T-GCN combines GCN and GRU to model complex non-Euclidean data with spatial and temporal dependencies [12]. GCN conducts convolution directly in the vertex domain, updating node representation by recursively aggregating neighbour information [23]. Messages flow between neighbours and the centre node to determine the architecture's propagation rules [24].The message propagation rules can be reworded into two stages— the message passing stage and readout stage, as shown in equation (1) as described in [24].

$$
\begin{aligned}
M_i^{l+1} &= \sum_{j\in N_i} Message(h_i^l, h_j^l, h_{ij}^l) \\
h_i^{l+1} &= Update(h_i^l, M_i^{l+1})
\end{aligned} \tag{1}
$$

h_i, h_j, and h_{ij} represent two specific nodes and their connecting edge feature. Node i's neighboring nodes are represented by the set N_i. The neighboring information is transformed into a hidden representation by the $Message()$ function and passed to the center node. $Update()$ function aggregates and updates the center node's representation. The choice of $Message()$ and $Update()$ functions may vary.

On the contrary, a GRU [25] is an improved RNN with reset and update gates that control information flow using sigmoid activation. The reset gate remembers the previous state, while the update gate copies the old state. Both gates use fully connected layers with sigmoid activation. As a result, T-GCN, the combination of GCN and GRU, significantly improves performance, making it the ideal choice for modeling spatiotemporal SCADA data.

E. T-GCN based AE

We have developed an AE model using the T-GCN layer described in the previous section (figure 1). The T-GCN layer replaces the layers of the conventional encoder and decoder architecture, enabling our AE to explicitly learn the spatiotemporal relationships of data. This unique feature makes the T-GCN layer-based AE more robust than GCN-based models. Initially, the input features are passed to the T-GCN layer of the AE, along with the edge index and edge weight of the adjacency matrix of the input graph-structured sensor data. We also added skip connections among the T-GCN layers to enhance our model further by reducing information loss during feature compression. We have used the MSE function as the loss function of the T-GCN AE. Let's consider the T-GCN encoder is represented as $\beta_e = E_{TGCN}(x, e_i, e_w)$, then the T-GCN decoder can be defined as $\bar{x} = D_{TGCN}(E_{TGCN}(x, e_i, e_w))$, where x, e_i, e_w, β_e and $\bar{x}$ are the input data, edges of the nodes, weights of the edges, encoded features and the reconstructed input data, respectively.

F. T-GCN AE based Fault Detection

This section explains the fault detection process using the T-GCN AE DL model for industrial environments. Data is collected from the SCADA network and divided into normal

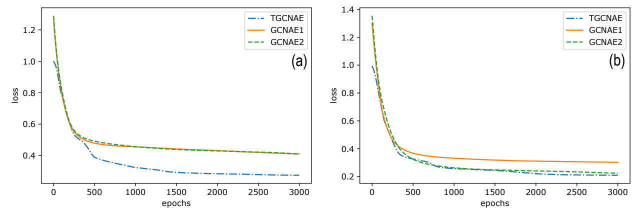

Fig. 3: Training losses of the models for WTB (a) and PVS (b) dataset.

and faulty parts. The normal and faulty data is processed using the FFT algorithm and outlier removal methods before training the T-GCN AE model to overcome the effects of environmental noise on data quality. The resulting cleaned data is used to create graph attributes. Following the procedure from GDA [8], we created the adjacency matrix for the graph. However, we used cosine similarity $\Big(CS(X,Y) = \frac{x.y}{||x||||y||}; where, x.y = \sum_i^n x_i y_i$ and $||x|| = \sqrt{x.x}\Big)$ instead of Euclidean distance to measure the closeness of neighbouring nodes [8]. We passed the cleaned normal data with the nodes' edge connectivity and weight values to T-GCN AE for training.

Then, we evaluated the trained model using validation data, a distinct set of normal data, and computed the squared prediction error (SPE) residuals $E \in R^{m\times n}$ of the validation data Δ_d and the reconstructed validation data $\tilde{\Delta_d}$ using equation (2).

$$SPE = (\Delta_d - \tilde{\Delta_d})^2; \Delta_d \in R^{m\times n} \tag{2}$$

To set thresholds for each variable in multivariate SPE $E \in R^{m\times n}$, we estimate their probability density functions using a non-parametric kernel density estimation (KDE) technique [26]. KDE is a successful method for processing monitoring and fault detection and is versatile in determining the threshold through the estimated PDF. The estimated PDF of some data points of a variable, say $x_i, i = 1, 2, .., N$ at point x can be defined as

$$p(x) = \frac{1}{N\Omega}\sum_{i=1}^{N}\kappa\Big(\frac{x - x_i}{\Omega}\Big) \tag{3}$$

Where, $\kappa(.)$ is the kernel function and Ω is the bandwidth. We have used the gaussian kernel function $\kappa(u) = \frac{e^{-\frac{u^2}{2}}}{\sqrt{2\pi}}$ and silverman bandwidth. Finally, the threshold (γ) of a variable can be computed using the variable's monitoring parameter (SPE) estimated PDF for a given confidence value α by solving [26] $P(x < \gamma) = \int_{-\infty}^{\gamma} p(x)\gamma(x) = \alpha$. During online monitoring phase, any data that surpasses the threshold (γ) can be defined as faulty samples. The fault detection process based on T-GCN AE is illustrated in figure 2.

IV. Experiments and Results

In this section, we discussed about the experimental procedure and outcomes from the experiments. Also, the performance of the T-GCN AE model is compared with the baseline models.

A. Dataset

To conduct the experiment, we have considered two dataset from wind turbine and photovoltaic system.

1) Wind turbine blade dataset (WTB): Wind turbines are prone to component failures due to irregular loads caused by wind turbulence and extreme weather [6]. This study analyzed vibration data from wind turbine blades operating under different load conditions, including healthy and faulty states such as blade erosion, mass imbalance, and cracked blades [27]. The dataset allows for assessing the impact of blade faults on power generation due to high vibration levels. The study considers only crack fault detection to validate the proposed model based on TGCN AE. The dataset's features are Wind Speed, Number of rotors, vibration measurement (X_T.D), T.D (Std Dev), T.D (Variance), T.D (Kurtosis), T.D (Skewness), T.D (RMS) Max, Crest factor, and Status.

2) PV system dataset (PVS): In a laboratory, a PV microgrid system is implemented to collect normal and faulty data to represent a Grid-connected PV system failure [28]. The data includes signals such as Time, PV array current, voltage, DC voltage, 3-phase current measurements, 3-phase voltage measurements, Current magnitude, Current frequency, Voltage magnitude, and Voltage frequency. Faults are introduced manually, including inverter faults, feedback sensor faults, grid anomalies, PV array mismatch, and controller and converter faults. This paper focuses on the inverter fault dataset to validate the proposed TGCN AE model for fault detection.

B. Baseline models

We have compared the GCN-AE [8] baseline model with our T-GCN AE model. We've made two versions of GCN-AE: GCN-AE1 uses Euclidean distance as similarity measurement technique without data processing, while GCN-AE2 uses cosine distance and applies data processing before training.

C. Experimental setting

1) Data processing: We removed noisy components from each feature of both datasets using the FFT algorithm with different thresholds based on visual inspection of the frequency domain plots. The resulting features were then subjected to the boxplot method to remove any outliers. However, this was insufficient for all features, so we utilized the IF method to suppress the remaining outlier points. We also discarded rows with missing values in this stage. We used 1000 training, 500 validation, 500 test samples from the WTB dataset and 2000 training, 500 validation, and 500 test samples from the PVS dataset. The models SPE monitored three features from the WTB dataset (TD_Std, TD_Var, TD_RMS) and one feature (V_pv) from the PVS dataset to detect faults in the wind turbine blade and microgrid. These features are chosen particularly because the faulty set of only these features shows high deviation from normal counterparts, as shown in figure 4 for the WTB dataset.

TABLE I: Performance comparisons of the models based on the performance metrics

Dataset	Models	train			valid			test		
		MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE
	T-GCN AE	0.2741	0.362	1.0801	0.5639	0.5701	1.3853	37.9286	3.8663	6.1743
	GCN-AE2	0.409	0.4892	5.4193	0.5833	0.6173	5.55	36.8733	3.9209	6.9573
WTB	GCN-AE1	0.4104	0.4445	3.5016	0.8177	0.7073	3.2018	2.0937	1.1845	2.5702
	T-GCN AE	0.2092	0.3420	0.9195	0.5294	0.5384	1.1930	4.8502	1.5002	1326.1900
	GCN-AE2	0.2239	0.3242	3.3280	0.4831	0.5056	4.3017	5.0450	1.6651	6.5788
PVS	GCN-AE1	0.3021	0.3228	3.0117	0.3300	0.3270	3.7799	2.2618	1.1126	4.9232

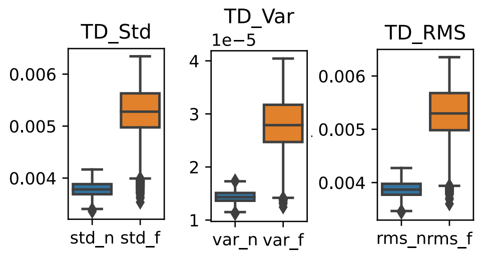

Fig. 4: Boxplot of the normal and faulty samples distribution for WTB dataset of features TD_{Std}, TD_{Var} and TD_{RMS}

2) Model training: Firstly, we normalized the processed dataset obtained from the previous stage using pythons $StandardScaler$ function $\Big((X_i - X_{mean})/X_{std}\Big)$. Next, we followed a procedure to generate the edge index and edge weights. The T-GCN AE and GCN-AE2 models use cosine distance, whereas GCN-AE1 uses Euclidean metrics for neighborhood similarity measurement. Then, we normalized the edge weights using the softmax function. Finally, we passed the processed data, the edge indices, and the edge weights to the models for training. The models were trained with 3000 epochs, a 0.01 learning rate, $Adam$ Optimizer and the MSE loss function for evaluation. Figure 3 displays the training MSE losses of all the models for both dataset.

3) Threshold computation: After training the model, we passed the validation dataset to the trained model and computed the SPE error for both datasets. The SPE for both datasets is calculated for each feature. Then, the thresholds are computed from the SPE using the Gaussian KDE estimation method following the equation (3) for confidence value $\alpha = 0.99$.

D. Results

1) Evaluation Metrics: We have considered three evaluation metrics– MSE, mean absolute error (MAE) and mean absolute percentage error (MAPE) to measure the data reconstruction ability of T-GCN AE, GCNAE1, and GCNAE2 models. The evaluation metrics are defined in equation (4), Where, N denotes the total number of data samples, yp and ya states the predicted data sample and actual (input) data samples respectively. The scores of all three models for MSE, MAE, and MAPE for training, validation, and test datasets can be observed in Table I. For WTB dataset, T-GCN AE outperforms GCN-AE1 and GCN-AE2 regarding data reconstruction ability for training and validation data splits. However, since the training and validation data come from the benign part of the dataset, the reconstruction error should be low compared to the test samples, which come from the faulty part of the dataset. Therefore, it can be seen that T-GCN AE has a higher MSE for the test data split than the training and validation split. While the MAE and MAPE of T-GCN AE are similar to the other models, it also has a higher MSE than GCN-AE1 and GCN-AE2 in test split. On the other hand, for PVS dataset, the data reconstruction ability of T-GCN AE is almost identical to GCN-AE1 and GCN-AE2. However, the abnormal data reconstruction ability of T-GCN AE2 for dataset 2 is still significantly higher than the other models, as reflected in the MAPE metrics.

$$\begin{aligned}
MSE &= \frac{1}{N}\sum_{i=1}^{N}(yp_i - ya_i)^2 \\
MAE &= \frac{1}{N}\sum_{i=1}^{N}|yp_i - ya_i| \\
MAPE &= \frac{100\%}{N}\sum_{i=1}^{N}|\frac{yp_i - ya_i}{ya_i}|
\end{aligned} \tag{4}$$

To determine the models performance in terms of fault detection capability and suppress false alarms, the FDR (tested on faulty part of SCADA data) and FAR (tested on benign part of SCADA data) are considered as defined in the following equations-

$$FDR = \frac{\lambda}{\Lambda} \tag{5}$$

where, λ denotes number of fault data that have been detected as fault and the Λ refers to total number of faulty samples.

$$FAR = \frac{\psi}{\Psi} \tag{6}$$

where, ψ stands for number of normal data that have been detected as fault and Ψ denotes total number of normal samples. Based on the results shown in table II, it is clear that the T-GCN AE and GCN-AE1 models have achieved a 100% FDR score, while the GCN-AE2 model has an FDR of 67.8%, 73.4%, and 65.8% for the three features of the WTB dataset, respectively. Although the T-GCN AE and GCN-AE1 models have similar FDR scores, the T-GCN AE model outperforms the other two regarding FAR. Furthermore, the T-GCN AE model also outperforms the other two models in detecting faults and generating false alarms in the PVS dataset.

TABLE II: Performance comparisons of T-GCN AE model against the GCNAE1 and GCNAE2 for FAR and FDR

Dataset	Models	Variables	FAR(%)	FDR(%)
WTB	T-GCN AE	TD_Std	0.2	100
		TD_Var	0.2	100
		TD_RMS	1.6	100
	GCN-AE1	TD_Std	4.6	100
		TD_Var	4	100
		TD_RMS	6.6	100
	GCN-AE2	TD_Std	6.4	67.8
		TD_Var	5.2	73.4
		TD_RMS	6.6	65.8
PVS	T-GCN AE	Vpv	0	100
	GCN-AE1	Vpv	59	100
	GCN-AE2	Vpv	70.6	91.6

V. Conclusion

This paper presents a highly effective AE-based fault detection model for wind turbine blades and photovoltaic microgrid dataset. Our proposed model incorporates a T-GCN layer to learn the spatiotemporal behavior of the components simultaneously, which is essential since SCADA data comprises spatial and temporal correlation among the features. While GCN-based AE models have been prevalent in recent studies for detecting faults by learning spatiotemporal data, our study demonstrates that the T-GCN-based AE model with noise reduction and outlier removal techniques yields more robust results with higher precision and fewer false alarms. Our experimental results show that the T-GCN-based AE model has achieved 100% FDR and minimal FAR for renewable energy industrial applications, making it a highly reliable and effective solution for fault detection. In our future work, we will develop a hybrid fault detection model by combining a GCN based forecasting and reconstruction model. Code will be made available to – https://github.com/ArifeenDipto/Fault-Diagnosis-Autoencoder

References

[1] H. Wang, Z. Lei, X. Zhang, B. Zhou, and J. Peng, "A review of deep learning for renewable energy forecasting," *Energy Conversion and Management*, vol. 198, p. 111799, 2019.

[2] G. Jiang, W. Li, W. Fan, Q. He, and P. Xie, "Tempgnn: A temperature-based graph neural network model for system-level monitoring of wind turbines with scada data," *IEEE Sensors Journal*, vol. 22, no. 23, pp. 22 894–22 907, 2022.

[3] X. Yu, B. Tang, and K. Zhang, "Fault diagnosis of wind turbine gearbox using a novel method of fast deep graph convolutional networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021.

[4] Z. Chen, J. Xu, T. Peng, and C. Yang, "Graph convolutional network-based method for fault diagnosis using a hybrid of measurement and prior knowledge," *IEEE transactions on cybernetics*, vol. 52, no. 9, pp. 9157–9169, 2021.

[5] J. Qian, Z. Song, Y. Yao, Z. Zhu, and X. Zhang, "A review on autoencoder based representation learning for fault detection and diagnosis in industrial processes," *Chemometrics and Intelligent Laboratory Systems*, p. 104711, 2022.

[6] E. S. Miele, F. Bonacina, and A. Corsini, "Deep anomaly detection in horizontal axis wind turbines using graph convolutional autoencoders for multivariate time series," *Energy and AI*, vol. 8, p. 100145, 2022.

[7] J. Liu, X. Wang, F. Xie, S. Wu, and D. Li, "Condition monitoring of wind turbines with the implementation of spatio-temporal graph neural network," *Engineering Applications of Artificial Intelligence*, vol. 121, p. 106000, 2023.

[8] L. Liu, H. Zhao, and Z. Hu, "Graph dynamic autoencoder for fault detection," *Chemical Engineering Science*, vol. 254, p. 117637, 2022.

[9] W. Wu, C. Song, J. Zhao, and G. Wang, "Knowledge-enhanced distributed graph autoencoder for multiunit industrial plant-wide process monitoring," *IEEE Transactions on Industrial Informatics*, 2023.

[10] T. Li, C. Suna, R. Yan, X. Chen, and O. Fink, "A novel unsupervised graph wavelet autoencoder for mechanical system fault detection," *arXiv preprint arXiv:2307.10676*, 2023.

[11] U. Goswami, J. Rani, H. Kodamana, S. Kumar, and P. K. Tamboli, "Fault detection and isolation of multi-variate time series data using spectral weighted graph auto-encoders," *Journal of the Franklin Institute*, vol. 360, no. 10, pp. 6783–6803, 2023.

[12] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[13] S. Yang, Y. Zhou, X. Chen, C. Li, and H. Song, "Fault diagnosis for wind turbines with graph neural network model based on one-shot learning," *Royal Society Open Science*, vol. 10, no. 7, p. 230706, 2023.

[14] Z. Lai, X. Cheng, X. Liu, L. Huang, and Y. Liu, "Multiscale wavelet-driven graph convolutional network for blade icing detection of wind turbines," *IEEE Sensors Journal*, vol. 22, no. 22, pp. 21 974–21 985, 2022.

[15] M. H. Raju, L. Friedman, T. Bouman, and O. Komogortsev, "Filtering eye-tracking data from an eyelink 1000: Comparing heuristic, savitzky-golay, iir and fir digital filters," *Journal of Eye Movement Research*, vol. 14, no. 3, Oct. 2023. [Online]. Available: https://bop.unibe.ch/JEMR/article/view/9888

[16] P. Anttonen, "Fourier transform techniques for noise reduction," 2022.

[17] M. H. Raju, L. Friedman, T. M. Bouman, and O. V. Komogortsev, "Determining which sine wave frequencies correspond to signal and which correspond to noise in eye-tracking time-series," *arXiv preprint arXiv:2302.00029*, 2023.

[18] J. Laurikkala, M. Juhola, E. Kentala, N. Lavrac, S. Miksch, and B. Kavsek, "Informal identification of outliers in medical data," in *Fifth international workshop on intelligent data analysis in medicine and pharmacology*, vol. 1, 2000, pp. 20–24.

[19] A. Smiti, "A critical overview of outlier detection methods," *Computer Science Review*, vol. 38, p. 100306, 2020.

[20] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*. IEEE, 2008, pp. 413–422.

[21] M. Arifeen, T. Ghosh, R. Islam, A. Ashiquzzaman, J. Yoon, and J. Kim, "Autoencoder based consensus mechanism for blockchain-enabled industrial internet of things," *Internet of Things*, vol. 19, p. 100575, 2022.

[22] M. Arifeen and A. Petrovski, "Bayesian optimized autoencoder for predictive maintenance of smart packaging machines," in *2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS)*. IEEE, 2023, pp. 1–6.

[23] M. Sun, S. Zhao, C. Gilvary, O. Elemento, J. Zhou, and F. Wang, "Graph convolutional networks for computational drug development and discovery," *Briefings in bioinformatics*, vol. 21, no. 3, pp. 919–935, 2020.

[24] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International conference on machine learning*. PMLR, 2017, pp. 1263–1272.

[25] A. Petrovski, M. Arifeen, and S. Petrovski, "Gated recurrent unit autoencoder for fault detection in penicillin fermentation process," in *International Conference on Intelligent Information Technologies for Industry*. Springer, 2023, pp. 86–95.

[26] G. Jiang, P. Xie, H. He, and J. Yan, "Wind turbine fault detection using a denoising autoencoder with temporal information," *IEEE/Asme transactions on mechatronics*, vol. 23, no. 1, pp. 89–100, 2017.

[27] A. A. F. Ogaili, A. A. Jaber, and M. N. Hamzah, "Wind turbine blades fault diagnosis based on vibration dataset analysis," *Data in Brief*, vol. 49, p. 109414, 2023.

[28] A. Bakdi, W. Bounoua, A. Guichi, and S. Mekhilef, "Real-time fault detection in pv systems under mppt using pmu and high-frequency multi-sensor data through online pca-kde-based multivariate kl divergence," *International Journal of Electrical Power & Energy Systems*, vol. 125, p. 106457, 2021.