*Advances in Reliability, Safety and Security, Part 9 – Kołowrocki, Dąbrowska (eds.)*
 *ISBN 978-83-68136-21-0 (printed), ISBN 978-83-68136-08-1 (electronic)*

**Advances in Reliability, Safety and Security**

ESREL 2024
Monograph Book Series

# Quantum Generative Adversarial Networks For Wind Turbine Shaft Vibration Prognostics

Aadittya Tiwari[a], Gabriel San Martín Silva[a],
Gustavo de Novaes Pires Leite[b], Enrique López Droguett[a],

[a]*Department of Civil and Environmental Engineering, University of California Los Angeles, Los Angeles, USA*
[b]*Federal Institute of Science, Technology, and Education of Pernambuco (IFPE), Recife, Brazil*

**Abstract**

In efforts to advance the development of clean energy sources, wind turbines have become increasingly popular for their ability to passively convert wind energy into electricity. Turbines require routine maintenance to remain operational, as there are many constantly moving components that degrade over time. To forecast when maintenance may be needed, turbine engineers use machine learning techniques such as time-series forecasting to predict the remaining useful life of the turbine. However, Quantum Machine Learning (QML), a rapidly developing field exploring quantum physics based hardware for running machine learning algorithms, is heavily unexplored in turbine prognostics and health management (PHM). This paper proposes a QML-based Generative Adversarial Network (GAN) specifically for forecasting vibrations within the shaft of a wind turbine. Our results show that compared to a classical GAN model, the Quantum GAN (Q-GAN) provides comparable performance and increased trainability, indicating a promising future for the development and application of QML in turbine PHM.

*Keywords*: prognostics & health management, reliability, quantum computing, machine learning, quantum machine learning, wind turbines, shaft vibrations, remaining useful life forecasting

## 1. Introduction

The increasing global demand for clean energy sources has led to the proliferation of clean energy sources over the course of the last century. One promising avenue for clean energy is the development of wind turbines to generate electricity. Turbines generate electricity by converting the kinetic energy of wind into mechanical energy through the rotation of blades, which then drives a generator to produce electrical power through electromagnetic induction. However, wind turbines are heavily dependent upon routine maintenance due to the inherent wear and tear of their constantly moving components leading to degradation. To combat this problem, reliability engineers can employ machine learning techniques, such as time-series forecasting, to forecast the level of degradation a particular turbine will experience in the future. This allows us to understand at exactly what point the turbine will reach a certain threshold for critical damage, and then let us to schedule proactive maintenance to repair degrading parts right when they begin to significantly affect the performance of the machine. While classical machine learning techniques have been proven to be effective for this purpose, Quantum Machine Learning (QML) remains almost completely unexplored in PHM despite its promising utility in the field.

Indeed, QML is a growing research topic as of late due to its potential in increasing program run times, optimization tasks, and machine learning model accuracy. While research on the applications of QML in various fields such as finance and chemistry are being thoroughly explored, applied research towards wind turbine PHM is almost non-existent. This paper aims to change that by putting forward a Quantum Machine Learning-based Generative Adversarial Network (Q-GAN) and utilizing it for the forecasting of harmful vibrations within the shaft of a wind turbine, which can then be used to determine when maintenance will be needed. Generative

Adversarial Network (GAN) models are known for their ability to produce synthetic data by using a Generator model within the GAN that tries to replicate observed trends in input data while a Discriminator model helps the Generator improve its performance. Currently, GANs are mainly used for tasks such as image generation. However, it can also be used for generating data matching the trends of input data such that we can theoretically use GANs for time-series forecasting tasks by inputting previous measurements of data and then getting an output of what the GAN believes future measurements of data will look like. GANs themselves have been used in time-series tasks (Wang et al., 2020) using GANs for short-term predictions of wind power output. While the results from (Wang et al., 2020) are reasonable, we theorize that the performance of GANs for these wind turbine PHM tasks can be improved with the use of a QML model as the generator inside a GAN instead of a classical generator, which would ultimately end up in powerful quantum models for turbine prognostics.

The paper is divided as follows. Section 2 presents an overview of the machine learning methods currently used in PHM. Section 3 introduces Quantum Computing and Machine Learning, while section 4 provides the construction and architecture of the proposed Quantum GAN. The processing of the turbine data is provided in Section 5, whereas Section 6 provides discussion on the results of the Q-GAN compared to a classical GAN for vibration prognostics. Section 7 offers some concluding remarks and directions for future work to build off the work of this paper.

## 2. Machine learning in prognostics and health management today

At the core of ML-based prognostics lies the utilization of vast datasets generated by sensors embedded in wind turbines. These sensors capture critical parameters such as rotor speed, temperature, vibration, and other operational metrics. ML algorithms, particularly supervised learning models, leverage historical data to discern patterns and anomalies, enabling the creation of predictive models for identifying potential issues. In (Wang et al., 2020), a semi-supervised GAN was used to extract non-linear and dynamic patterns from the wind energy data to perform prediction of wind power in the future. Another example would be the use of Variational Autoencoders to perform compression over large files of ball-bearing fault data (San Martin et al., 2018), which can greatly reduce the time-complexity involved with classification tasks of the ball bearing elements, as there is simply less data for the model to look over. A final use case would be the usage of a LSTM-CNN model for to predict the energy output of turbines in an off-shore wind farm based on historical data and information of the turbines (Chen et al., 2021). These examples provide a basis for the implementation of machine learning models in prognostics and health management.

## 3. A brief introduction to quantum computing and machine learning

Quantum computing is an emerging field which promises to revolutionise how we solve complex problems. Unlike classical computers, which rely on bits that can only exist in one of two states, quantum computers use quantum bits, or qubits, which are represented as probabilities of being a 0 or 1 - it is not definite. This is an example of quantum superposition, and with other properties such as quantum entanglement between qubits, qubits become powerful as computational devices. These properties can be exploited to perform certain calculations much faster than classical computers, making them particularly useful for tasks such as machine learning. The python library used in this paper to run quantum machine learning algorithms is Pennylane (Bergholm, Izaac and Schuld, 2022).

Machine learning (ML) models are algorithms which learn from data without being explicitly programmed. It involves training models to identify patterns in data and make predictions or classifications based on those patterns. Training a ML model consists of three main stages: data preparation, model training, and model evaluation. During the data preparation stage, data is collected, cleaned, and relevant features are extracted or engineered for the model to be trained on. In the model training stage, machine learning algorithms are used to train models on the prepared data. Finally, in the model evaluation stage, the trained models are tested on new data to ensure their accuracy and effectiveness. If the model currently is not performing to the level of accuracy which you would expect on given data, we go back to the training stage and keep repeating the process until the model works properly.

## 4. Quantum-GAN architecture

This paper proposes a Quantum GAN (Q-GAN), which is a hybrid machine learning model that combines quantum and classical computing. The generator for the Q-GAN is a Quantum Long Short-Term Memory circuit (QLSTM) (Chen, Yoo and Fang, 2020). The QLSTM replaces a layer in a classical LSTM circuit with 6 Variational Quantum Circuits (VQCs) to form a QLSTM cell. VQCs are a type of quantum circuit used in quantum machine learning and optimization, which are parameterized by a set of randomly initialised parameters and optimised to minimise a cost function related to the task objective. VQCs have the advantage of being able to represent complex functions using a small number of qubits and gates, thus theoretically providing computational advantages over classical machine learning algorithms. This QLSTM generator uses 4 qubits from the Pennylane's default.qubit Quantum Computer simulator. As shown later, the new quantum layer in the proposed model was found to have quicker convergence and a more stable loss function graph than its classical counterpart. To recap the differences, the information flows from which data is processed between LSTMs and QLSTMs is shown below.

The information flow in an LSTM is as following:

$$f_t = \sigma\left(W_f \cdot v_t + b_t\right) \tag{1}$$

$$i_t = \sigma(W_t \cdot v_t + b_i) \tag{2}$$

$$C_t = \tanh(W_c \cdot v_t + b_c) \tag{3}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot C_t \tag{4}$$

$$o_t = \sigma(W_o \cdot v_t + b_o) \tag{5}$$

$$h_t = o_t \cdot \tanh(c_t), \tag{6}$$

where $\sigma$ denotes the sigmoid functions, $\{W_n\}$, are classical neural networks ($n = f, i, C, o$), where $f$ represents the forget block, $i$ represents the input block, $C$ represents a new state cell candidate, and $o$ represents the output block.

The architecture of the LSTM can be visualised with the following image.

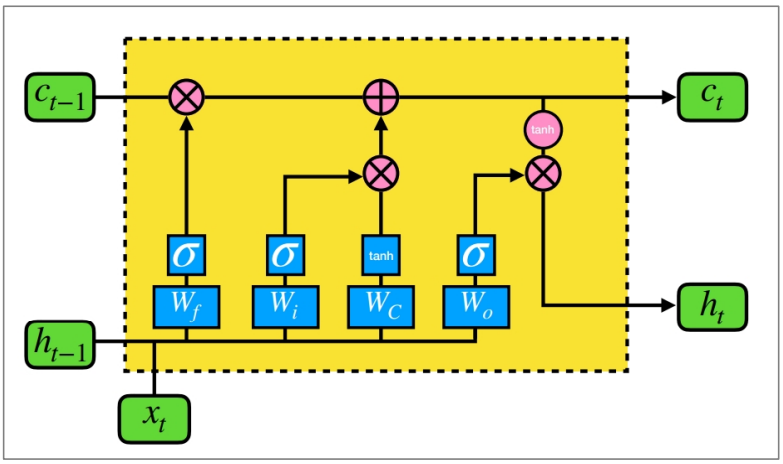


Fig.1. Classical LSTM architecture.

While this model works well for time-series, we attempt to further improve its performance with the use of Variational Quantum Circuits within the information flow. The information flow in a Quantum LSTM can be modelled as:

$$f_t = \sigma\left(VQC_1(v_t)\right) \tag{7}$$

$$i_t = \sigma\left(VQC_2(v_t)\right) \tag{8}$$

$$C_t = \tanh\left(VQC_3(v_t)\right) \tag{9}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot C_t \tag{10}$$

$$o_t = \sigma\left(VQC_4(v_t)\right) \tag{11}$$

$$h_t = VQC_5(o_t \cdot \tanh(c_t)) \tag{12}$$

$$y_t = VQC_6(o_t \cdot \tanh(c_t)), \tag{13}$$

where the term *VQC* denotes various Variational Quantum Circuits that will be used in the hybrid Quantum LSTM. Note that the set of equations are alike between the classical LSTM and the Quantum LSTM. However, the main difference, and proposed advantage, comes from the usage of the VQCs in the Quantum LSTM. The architecture of the QLSTM can be visualised with the following image.

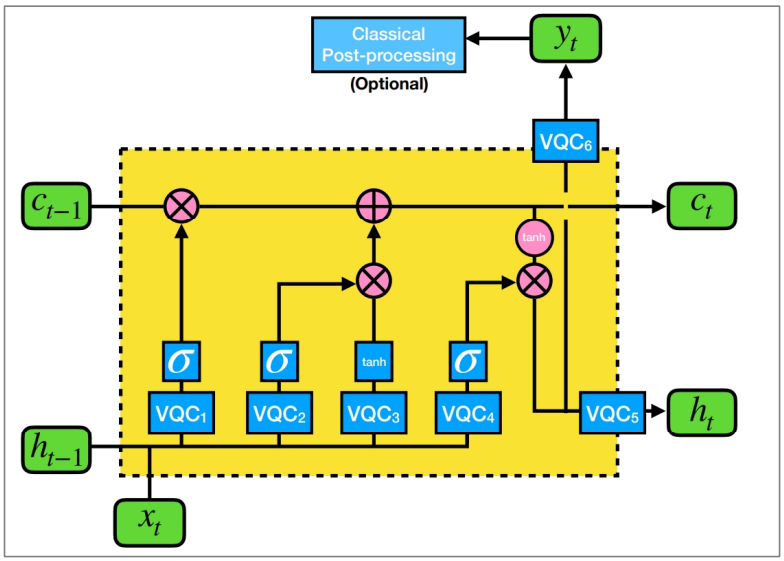


Fig. 2. Quantum LSTM architecture.

Each VQC box is integrated in the architecture as detailed in Figure 4. The $\sigma$ and tanh blocks represent the sigmoid and the hyperbolic tangent activation function, respectively; $i_t$ is the input at time $t$, $h_t$ is for the hidden state, $c_t$is for the cell state, and $y_t$ is the output; $\otimes$ and $\oplus$ represents element-wise multiplication and addition, respectively.

## 5. Processing Turbine Data

The dataset is collected from a 2MW wind turbine with high-speed shaft driven by a 20-tooth pinion gear. A vibration signal of 6 seconds was acquired each day for 50 consecutive days (there are 2 measurements on March 17, which are treated as two days in this example). An inner race fault developed and caused the failure of the bearing across the 50-day period (The MathWorks Inc., 2024). The dataset can be found at the link: https://github.com/mathworks/WindTurbineHighSpeedBearingPrognosis-Data. We start by converting the file types to pandas dataframes for easy data manipulation, allowing us to down sample the dataframe in favour of time-efficiency as well as engineering statistical features from the vibration data to help understand patterns from the data. The dataset now contains 29,198 x 10 data points. Naturally, this volume leads to high training times as well as the curse of dimensionality (Keogh and Mueen, 2017). To address both these issues, we train and use an Autoencoder consisting of fully connected linear layers. The encoder (E) transforms the $n$ x $m$ dataset $X$ into a lower-dimensional latent space representation:

$$Z \in \mathbb{R}(n \cdot 1) : Z = E(X). \tag{14}$$

The decoder ($D$) then reconstructs the original dataset from the latent space representation:

$$X = D(Z). \tag{15}$$

The loss function used to measure accuracy is the Mean Squared error between the real values inputted into the autoencoder and the reconstructed value outputted from the decoder. During training, the Mean Squared Error loss is minimized between the original data $X$ and the reconstructed data $\hat{X}$:

$$L(X, X_o) = \sum_i \sum_j \left(X_{ij} - X_{o_{ij}}\right), \tag{16}$$

where $i$ ranges from 1 to $n$, and $j$ ranges from 1 to $m$. The goal is to train the autoencoder to learn a representation of the original data in the lower-dimensional latent space ($n$ x $1$). This solves both the problems of high training times and the curse of dimensionality. We now have a $n$ x $m$ dataset's information in a more manageable $n$ x $1$ latent space representation to train the GAN. The GAN will forecast the latent space, the forecasted latent space can then be reconstructed with the decoder of the autoencoder, which then leads to the forecasts of the vibration to be used for turbine health evaluation.

## 6. Results

With our data processed, we are now ready to train the GANs and Q-GANs. To standardise both the models as much as possible, we will use Bayesian optimization (Snoek, 2012) to select the most optimal hyperparameters (generator and discriminator learning rates, batch size, epoch count, and betas values) for each model. After training, we will compare the results of both the models for forecasting. Metrics used for evaluation will be forecast RMSE compared to training values, real values, epochs taken to converge, total model parameters, and time taken to run. Results are in the table below.

Table 1. Results.

| Model | Classical GAN | Quantum GAN |
|---|---|---|
| Training RMSE | 5.689 | 6.950 |
| Testing RMSE | 9.933 | 10.436 |
| Epochs taken to converge | 48 | 19 |
| Total Model Parameters (Gen. + Disc.) | 8186113+ 109429 | 169+109429 |
| Time taken to run | ~0.46 hours | ~12.79 hours |

We have also included images of the forecasts versus the real values in the testing and training datasets for each model to help visualise the accuracy the models.

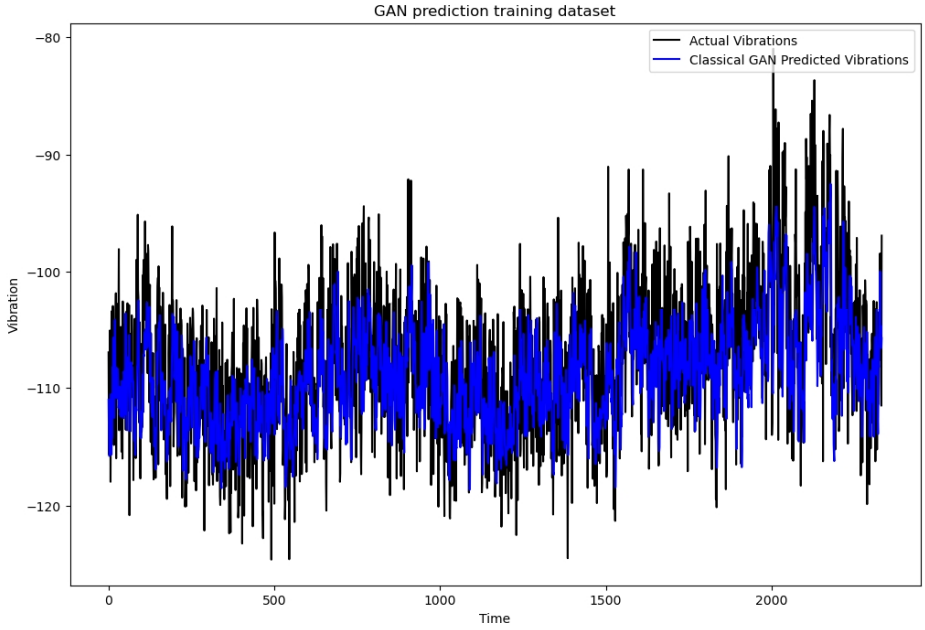


Fig. 3. Classical GAN prediction on the training dataset.

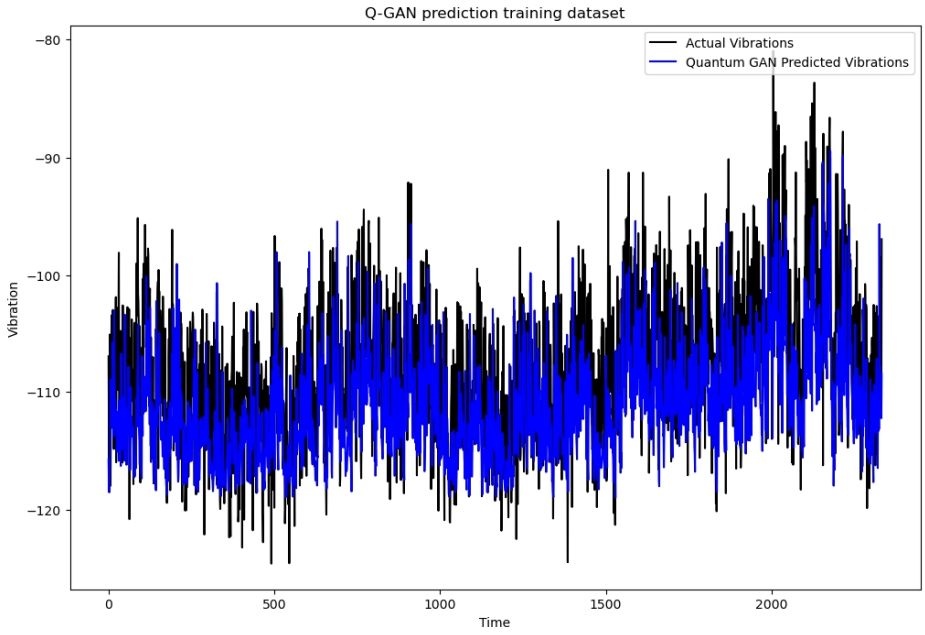


Fig. 4. Quantum GAN prediction on the training dataset.

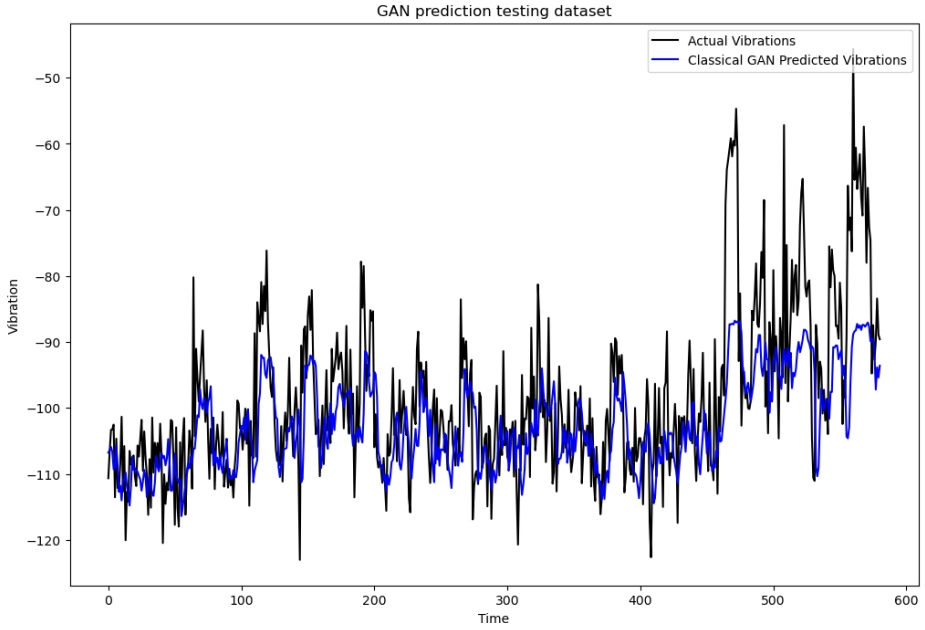


Fig. 5. Classical GAN prediction on the testing dataset.

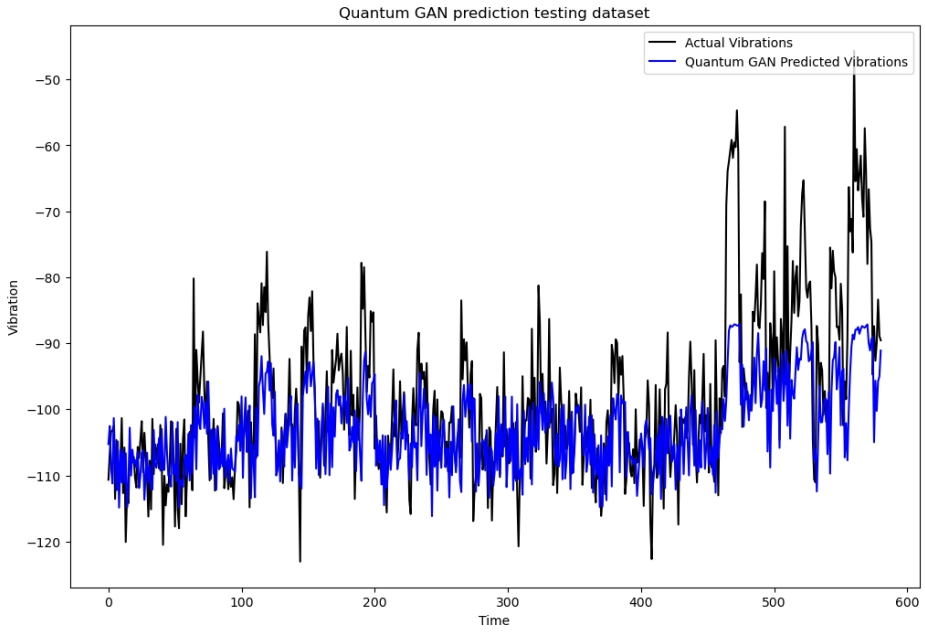


Fig. 6. Quantum GAN prediction on the testing dataset.

## 7. Conclusions and future research

Based on the above results, the proposed Quantum GAN was able to provide a comparable performance when compared to the classical GAN. Note also that the Quantum GAN was able to converge much quicker than the classical GAN, with much fewer total parameters as well. In fact, near the end of the graphs for Figures 3 and 4, the Quantum GAN was able to catch onto a large spike in the vibrations, whereas this behaviour was missed by the classical GAN. Being able to catch transients indicative of faults, which translates to abnormal events where a component has severely broken instantly, is what one needs to detect and model to be able to develop efficient and reliable maintenance strategies. Regarding the trainability of the models, the usage of the Variational Quantum Circuits as trainable models enabled the Quantum GAN to essentially "train more in one epoch than the Classical GAN does in one epoch". It is also important to note that there are significantly less parameters in the Quantum GAN, which can be attributed to only needing to tune qubit angles within the Quantum LSTM, as compared to a greater variety of tuneable parameters in the classical GANs layers. Finally, because of using a simulator (a classical computer coded to behave like a quantum computer) and not a real quantum computer, it will take you more time to train quantum models unless one uses a real quantum computer. While classical epochs could be trained in under 20 seconds, epochs for quantum epochs took around 10 minutes each. Possible room for improvement would be to try out different Quantum Computing simulators from different providers, or in the future evaluate performance on a real quantum computer. As hardware scales with time, so will algorithms making Quantum Machine Learning models, such as the proposed Quantum GAN, far more effective and pave the way for algorithms that are not constructed from existing classical machine learning algorithms and are instead unique with no classical counterparts. Overall, the Quantum GAN has shown itself to be a competitive counterpart to classical GAN in terms of performance and further shows the future of Quantum Machine Learning as a viable alternative to Classical Machine Learning for prognostics and health management tasks.

## References


Bergholm, V., Izaac, J., Schuld, M. 2022, July 29. Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv.org. https://arxiv.org/abs/1811.04968

Chen, S. Y.-C., Yoo, S., Fang, Y.-L. L. 2020, September 3. Quantum long short-term memory. arXiv.org. https://arxiv.org/abs/2009.01783

Chen, X., Zhang, X., Dong, M., Huang, L., Guo, Y., He, S. 2021, July 16. Deep learning-based prediction of wind power for multi-turbines in a wind farm. Frontiers. https://www.frontiersin.org/articles/10.3389/fenrg.2021.723775/full

Keogh, E., Mueen, A. 2017. Curse of dimensionality. In Springer eBooks, 314-315. https://doi.org/10.1007/978-1-4899-7687-1_192

San Martin, G., López Droguett, E., Meruane, V., das Chagas Moura, M. 2018. Deep variational auto-encoders: A promising tool for dimensionality reduction and ball bearing elements fault diagnosis. Structural Health Monitoring 18(4), 1092-1128. https://doi.org/10.1177/1475921718788299

Snoek, J. 2012, June 13. Practical Bayesian optimization of machine learning algorithms. arXiv.org. https://arxiv.org/abs/1206.2944

The MathWorks, Inc. 1994-2024. Wind Turbine High-Speed Bearing Prognosis - MATLAB & Simulink. https://www.mathworks.com/help/predmaint/ug/wind-turbine-high-speed-bearing-prognosis.html (Accessed 01.05.2024).

Wang, H., Gallego-Castillo, C., Koivisto, M., Hong, D. Y., Zuluaga, C. D., Wang, Y., Fei, S., Zhang, Y., Li, Z., Cao, Q., Liu, H., Dhiman, H. S., Da, X. U. 2020, September 10. Short-term prediction of wind power and its ramp events based on semi-supervised Generative Adversarial Network. International Journal of Electrical Power & Energy Systems. https://www.sciencedirect.com/science/article/abs/pii/S0142061520303586?via%3Dihub