Autoencoder based Semi-Supervised Anomaly Detection in Turbofan Engines

Ali Al Bataineh[1], Aakif Mairaj[2], Devinder Kaur[3]
EECS Department, College of Engineering
University of Toledo, Toledo
OH, 43606

***Abstract*—This paper proposes a semi-supervised autoencoder based approach for the detection of anomalies in turbofan engines. Data used in this research is generated through simulation of turbofan engines created using a tool known as Commercial Modular Aero-Propulsion System Simulation (CMAPSS). C-MAPSS allows users to simulate various operational settings, environmental conditions, and control settings by varying various input parameters. Optimal architecture of autoencoder is discovered using Bayesian hyperparameter tuning approach. Autoencoder model with optimal architecture is trained on data representing normal behavior of turbofan engines included in training set. Performance of trained model is then tested on data of engines included in test set. To study the effect of redundant features removal on performance, two approaches are implemented and tested: with and without redundant features removal. Performance of proposed models is evaluated using various performance evaluation metrics like F1-score, Precision and Recall. Results have shown that best performance is achieved when autoencoder model is used without redundant feature removal.**

Keywords*—Anomaly detection; autoencoder; bayesian hyperparameter tuning; turbofan engine

I. Introduction

Anomaly detection refers to identification of those situations, which do not conform to pre-defined normal behavior of the system under consideration. Timely detection of such anomalies in machinery has many applications, which include reduced downtime, reduced maintenance cost and less safety hazards. Increasing focus on reliability and maintenance of complex systems like turbofan engines demands intelligent and autonomous ways to manage the health of these safety critical systems [1]. One such way is to deploy autonomous anomaly detection to monitor the health of turbofan engines. Timely detection of anomalies in turbofan engines can enable its operators to take corrective actions timely and prevent catastrophic failures.

In recent years, there is an increasing interest in data driven anomaly detection techniques. Based on nature of available dataset, data driven anomaly detection techniques are classified into three types: supervised, unsupervised, and semi supervised [2].

Supervised anomaly detection techniques require true labels for all training instances: normal as well as anomalous. Bayesian networks in supervised setup are used for intrusion detection in [3].

Researchers have also used multilayer perceptron (MLP) for detection of normal and attack connections [4]. In addition to detection of attack connections, MLP based approach also helps to identify type of attacks. Researchers have also used support vector machines (SVM) and decision trees for detection of anomalies in various applications [5], [6]. One of the biggest challenges in supervised anomaly detection approaches is non availability of representative labels, especially for anomalous class [7]. Another issue is that anomalous class has far fewer instances than normal class instances. This issue of imbalance class distribution is addressed in [8].

Unsupervised anomaly detection algorithms do not require true labels of data instances for training. Basic assumption for these algorithms is that only small percentage of data belongs to anomalous class. Unsupervised anomaly detection approaches classify infrequent data instances as anomalous [2]. K-means clustering based sliding window approach is used to detect anomalies in discrete manufacturing process by [9].

Another clustering algorithm called Fuzzy C-Means (FCM) is also used by researchers for unsupervised anomaly detection [10]. In FCM, one data instance can belong to more than one clusters.

Some researchers have also used expectation maximization (EM) meta algorithm for unsupervised anomaly detection [11]. EM is again a soft clustering technique which maximizes the value of certain parameter in a probabilistic model. One of the biggest issues with these clustering-based anomaly detection techniques is high false positive rate.

In [12], different clustering algorithms were used for intrusion detection and it was observed that false positive rate was more than 20%. Such high false positive rate makes clustering based anomaly detection techniques unviable for some of the real-world problems [12].

Another challenge in unsupervised anomaly detection techniques is the assumption that only small proportion of data represents the anomalous class. In situations where this assumption is not true, these unsupervised anomaly detection approaches may suffer from bad performance.

Semi-supervised anomaly detection approaches require that true labels should be available for only those data instances which represent the normal behavior of system under consideration. As these semi-supervised approaches do not require the labels for anomalous data instances, therefore these approaches are widely applicable in practice as compared to supervised anomaly detection approaches.

In [13], a kernel principal component analysis (PCA) based approach is used to deploy semi-supervised anomaly detection on a spacecraft. This approach first projects original data on to a lower dimensional space and then reconstructs it from that lower dimensional space. This reconstructed data from lower dimensional space is supposed to represent the true nature of data (independent of noise). The reconstruction error between original data and reconstructed data is then used to detect anomalies. However, the performance of this approach is highly dependent upon type and hyperparameters (e.g., degree in case of polynomial kernel) of kernel chosen. In some cases, choice of kernel requires domain knowledge, which is not easily available in all the cases.

In [14], researchers have proposed a semi-supervised support vector machines (S3VMs) for detection of anomalies in complex systems. It has been observed that semi supervised SVMs give high false positive rate when tuned and trained in a semi-supervised setup [15]. In addition to this, curse of dimensionality is also an issue when these SVMs are used with high dimensional data. In recent years, there is an increasing interest in use of artificial neural networks for various applications. Autoencoder is also a type of neural network, which is designed to learn reconstruction of input data [16].

Unlike PCA based approaches, autoencoders perform hierarchical dimensionality reduction by stacking up multiple hidden layers. By reducing number of neurons in subsequent hidden layers, each hidden layer tends to learn the true nature of the data. So, by using multiple hidden layers in auto-encoder framework, more abstract features can be extracted, and better reconstruction of data can be achieved without any dependency on domain knowledge.

In this research, an autoencoder based semi-supervised anomaly detection approach is used to detect anomalies in turbofan engines. As explained earlier, the main advantage of using autoencoders for anomaly detection is that they require only normal data for training and their performance is also not dependent on any user defined parameters (e.g., kernel type).

The rest of the paper is organized as follows. In section II, adopted methodology is explained in detail. Dataset used in this research is explained in section III. Section IV contains the implementation details and results. Conclusion of this research is presented in section V.

II. Methodology

As stated earlier that an autoencoder based anomaly detection approach is used in this research. A detailed explanation of the adopted methodology is given in this section.

A. Autoencoders

Autoencoder is an artificial neural network, which is trained to learn the reconstruction of input signal. Internally, an autoencoder consists of two parts: encoder and decoder. First of all, input signal $x \in [0,1]^d$ is mapped to a hidden representation $y \in [0,1]^{d'}$ through an encoding function $f(x)$. This hidden representation y is also known as the code of autoencoder. Here d and d^0 represents the dimensions of x and y respectively. Hidden representation y or code is then mapped back to reconstruction $z \in [0,1]^d$ through decoding function $g(y)$. The dimension of reconstruction z is same as of x. Here z should be considered as prediction of x by an autoencoder having code y. This structure of autoencoder is presented in Fig. 1.

Mathematical expressions of encoder and decoder functions are presented in in Eq. (1) and Eq. (2), respectively.

$$y = f(x) = \sigma(W_{xy}x + b_{xy}) \quad (1)$$

$$y = g(y) = \sigma(W_{yz}y + b_{yz}) \quad (2)$$

Here W represents the weight, b represents the bias and represents the nonlinear activation function of neural network. Learning process of an autoencoder involves the minimization of loss between x and $g(f(x))$. Loss function used in this research is the mean squared error (*MSE*). This loss function L is presented in Eq. (3).

$$L\left(x, g\left(f(x)\right)\right) = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - g\left(f(x_i)\right)\right)^2 \quad (3)$$

where n represents total number of training examples.

An autoencoder which learns to perfectly reconstruct x everywhere (for all values of x) is not generally useful. Therefore, autoencoders are generally designed in such a way that they can perfectly reconstruct only those inputs which resemble to data in training set. One way to restrict perfect reconstruction everywhere is to constraint code y to have lower dimension than x. Type of autoencoder in which dimension of x is greater than the code y (i.e. $d > d^0$) is known as undercomplete autoencoder [17]. In this research we have used an undercomplete autoencoder to build an anomaly detection model.

In this paper, we have trained our autoencoder model on data representing normal behavior of system under consideration. A perfect autoencoder which is trained on normal data should be able to reconstruct only those inputs which are representative of normal behavior of system under consideration. Metric which is used to quantify the quality of reconstruction is reconstruction error. Reconstruction error can be measured in many ways. In this research we have used sum of squared error between x and z to measure the reconstruction error. This is presented in Eq. (4).

$$\text{reconstruction error (r)} = \sum_{i=1}^{k}(x_i - z_i)^2 \quad (4)$$

where k represents the dimension of input signal.

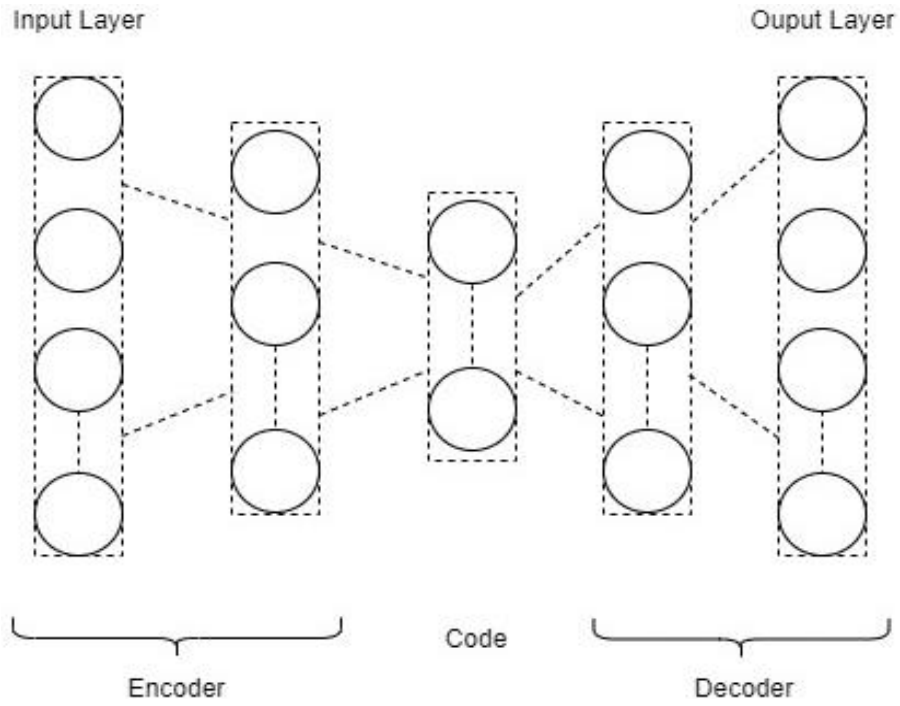


Fig. 1. An Autoencoder Structure.

An autoencoder which is trained on normal data should have a smaller reconstruction error on datapoints which are representative of normal behavior of the system and vice versa. Hence anomalies can be detected by simply using a threshold on reconstruction error. Data points having reconstruction error less than a certain threshold can be classified as normal, whereas data points having reconstruction error greater than a certain threshold can be classified as anomalies. This is presented in Eq. (5).

$$\begin{cases} r > Th\ Anomaly \\ r < Th\ Normal \end{cases} \quad (5)$$

where *Th* represents the threshold value.

The performance of an autoencoder model is highly dependent on the choice of hyperparameters, such as the number of layers, number of neurons and activation function, etc. The approach adopted for hyperparameter tuning in this research is Bayesian optimization, which is explained in the following section.

B. Hyperparameter Tuning using Bayesian Optimization

The aim of the hyperparameter tuning task is to find the set of hyperparameters, which gives the best performance (e.g., F1-score, R2-score, etc.) on the validation dataset for a specific model [18], [19].

For complex models like neural networks, manual tuning of these hyperparameters becomes intractable. There are some approaches like grid search and random search, which perform better than manual search in most of the cases. In the grid and random search, a search grid is being set up and the train-predict-evaluate cycle is executed for a different set of hyperparameters in a loop. However, these approaches are inefficient in the sense that they do not consider the performance of previously chosen hyperparameters while choosing the next set of hyperparameters. Grid and random search will continue to search the whole search space while being uninformed about the past evaluations. As a result, an ample amount of time is usually spent on the evaluation of bad hyperparameters.

In contrast to the grid and random search, the Bayesian approach for hyperparameter tuning considers past evaluations' results while choosing a new set of hyperparameters [20].

There are multiple approaches for Bayesian optimization in literature, differentiated based on the type of regression model and acquisition function they use. A probabilistic regression model is used to model the past evaluations by mapping hyperparameters to score on objective function. This regression model is also known as surrogate model in literature and is represented as $p(s/h)$ [20]. Here s represents the score on objective function and h represents the set of hyperparameters. Whereas next set of hyperparameters (from domain) in each iteration is chosen by optimizing an acquisition function, which uses $p(s/h)$ as a cheap surrogate of actual objective function.

In this work, we have used Tree-Structured Parzen Estimator (TPE) to build the surrogate model of objective function. Tree-Structured Parzen Estimator builds the surrogate model by using Bayes rule. Instead of directly calculating $p(s/h)$, it calculates $p(h/s)$ first and then use Bayes rule as in Eq. (6).

$$p\left(\frac{s}{h}\right) = \frac{p\left(\frac{h}{s}\right) * p(s)}{p(h)} \quad (6)$$

where $p(h/s)$ is probability of hyperparameter given the score and is expressed as in Eq. (7).

$$p\left(\frac{h}{s}\right) = \begin{cases} l(h) \; if \; s < s^* \\ g(h) \; if \; s \geq s^* \end{cases} \quad (7)$$

In Eq. (7), hyperparameters are divided into two distributions: $l(h)$ and $g(h)$. $l(h)$ contains all those set of hyperparameters for which score(s) of objective function is less than a certain threshold s^*, whereas g(h) contains all those set of hyperparameters, for which score(s) of objective function is greater than a certain threshold s^*. Acquisition function used in this research is Expected improvement (EI). The main task of the acquisition function is to find best set of hyperparameters based on surrogate model $p(s/h)$. Mathematical expression of Expected Improvement (EI) is given in Eq. (8).

$$EI_{s^*}(h) = \int_{-\infty}^{s^*} (s^* - s) p(d/h) ds \quad (8)$$

III. Case Study

In this work, we picked up a case study of anomaly detection in a simulated dataset of a turbofan engine [21]. The first principle model required to generate the data is built using a tool known as Commercial Modular Aero-Propulsion System Simulation (C-MAPSS).

C-MAPSS allows users to simulate various operational settings, environmental conditions, and control settings by varying various input parameters. In the chosen dataset, there is run-to-failure data of 249 engines simulated under six different operational settings. Some manufacturing variations and different initial degree of wear are being introduced in all 249 engines in order to make the scenario more real. Initial wear in engines is being introduced by varying efficiencies of various modules. In all the engines, a fault is introduced due to either of two failure modes: High Pressure Compressor (HPC) Degradation and Fan Degradation. At the start of each time series, the engine is running in a normal state and fault is introduced at some point in time, which then leads to engine failure in the future.

The Health state of each engine is measured by a set of 21 sensors installed on different modules of the engine. A list of all sensors is presented in Table I. In addition to these 21 sensor tags, three additional parameters are recorded to represent different operating states of the engine. A list of operational parameters is presented in Table II. The values of all these sensors and operational parameters are recorded at a frequency of one reading per engine cycle.

For semi-supervised anomaly detection, we are required to train our machine learning model on data representing the normal behavior of the system under consideration. As explained earlier, at the start of each time series, all engines are operating in a normal state, therefore in this work, the first 60 percent data of each time-series is considered as representative of the normal behavior of engines. The threshold of 60 percent is decided based on visual analysis of trends. Out of 249 engines, the first 60 percent data of 220 randomly chosen engines is used for training. Data of 20 engines is used as validation data (for hyperparameter tuning), and data of the remaining 19 engines is used for testing the performance of the trained model.

TABLE I. Sensors Names and their Units

Sensor	Description	Unit of Measure
T2	Fan inlet temperature	Rankine (°R)
T24	Low Pressure Compressor (LPC) outlet temperature	Rankine (°R)
T30	HPC outlet temperature	Rankine (°R)
T50	Low Pressure Turbine (LPT) outlet temperature	Rankine (°R)
P2	Fan inlet pressure	Pounds Per Square Inch Absolute (PSIA)
P15	Bypass-duct pressure	Pounds Per Square Inch Absolute (PSIA)
P30	HPC outlet pressure	Pounds Per Square Inch Absolute (PSIA)
Nf	Fan speed	Revolution Per Minute (rpm)
Nc	Core speed	Revolution Per Minute (rpm)
Epr	Engine Pressure Ratio	Nil
Ps30	HPC outlet static pressure	Pounds Per Square Inch Absolute (PSIA)
Phi	Fuel flow ratio to Ps30	pps/psi
NRF	fan corrected speed	Revolution Per Minute (rpm)
NRc	Core corrected speed	Revolution Per Minute (rpm)
BPR	Bypass ratio	Nil
farB	Fuel-air ratio of burner	Nil
htBleed	Bleed enthalpy	Nil

TABLE II. Operational Parameters

Operational Parameter	Description
Tr	Throttle Resolver Angle (TRA)
Al	Altitude
MN	Match Number

IV. Implementation

As detailed earlier, in this work, we have used semi-supervised autoencoders for detecting anomalies in turbofan engines. The overall approach can be divided into two phases: training and testing. In the training phase, training data representing the normal behavior of the engines is used to train the optimal autoencoder model. The effect of the removal of redundant features on model performance is evaluated using Pearson's correlation. If multiple features are found correlated with each other, only one is used in model training/testing. Features selected through this approach are listed in Table III. In this research, the results of both approaches (with and without redundant features removal) are presented.

The optimal architecture of autoencoder for given training data is discovered using the Bayesian optimization-based hyperparameter tuning approach. Search ranges of all the hyperparameters which are tuned using Bayesian optimization are given in Table IV. These search ranges are the same for both approaches (with and without redundant features removal). The final set of hyperparameters discovered by Bayesian hyperparameter tuning for both approaches is

presented in Table V. After figuring out the optimal architecture of the autoencoder; the following task is to train the autoencoder on normal training data. This is achieved using the backpropagation algorithm [22, 23].

TABLE III. FEATURES SELECTED AS RESULT OF REDUNDANT FEATURE REMOVAL

Feature Name	Feature Type
Throttle Resolver Angle	Operational Parameter
Altitude	Operational Parameter
Fan inlet temperature	Sensor
Engine Pressure Ratio	Sensor
HPC outlet static pressure	Sensor
Bypass ratio	Sensor

TABLE IV. HYPERPARAMETERS SEARCH RANGE FOR AUTOENCODER

Hyperparameter	Range
Number of Epochs	1-127
Batch Size	1-256
Number of Layers	[3, 5, 7, 9, 11]
Activation Function	Sigmoid, Softmax, Tanh, ReLU
Optimizer	Adam, Adadelta, RMS, SGD

TABLE V. FINAL HYPERPARAMETER SET

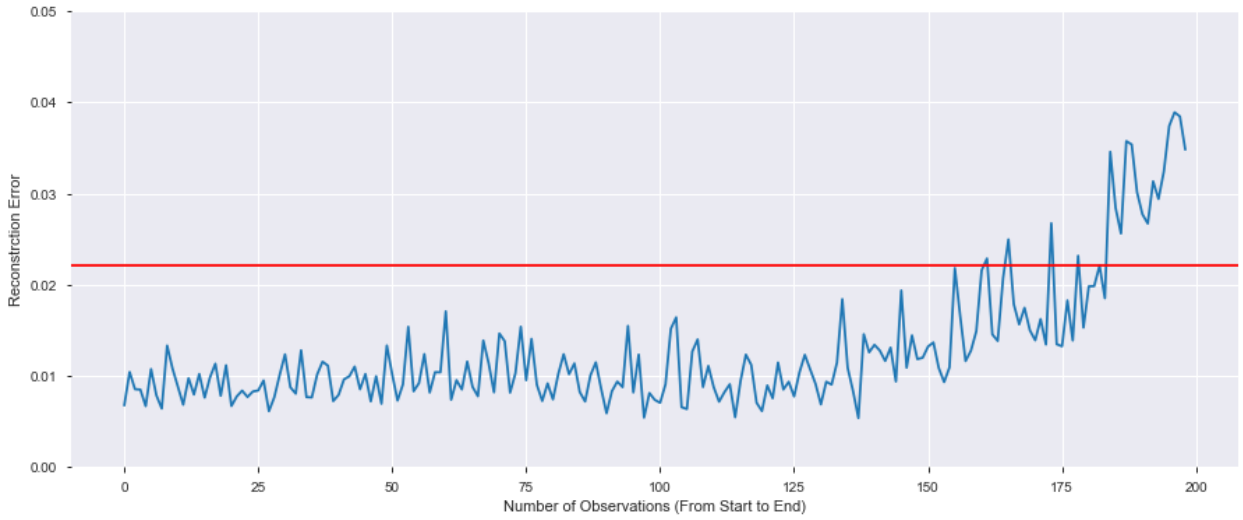
Hyperparameter	Without Feature Removal	With Feature Removal
Number of Epochs	64	22
Batch Size	148	10
Number of Layers	5	3
Number of Neurons per Layer	[24, 12, 8, 12, 24]	[6,3,6]
Optimizer	Adam	RMSprop

Once autoencoder is trained, next task is to compute the threshold value which is required for detection of anomalies during testing phase. For threshold calculation, all training samples (which are representative of normal behavior of the system) are scored through trained autoencoder model. For all scored samples, reconstruction error is computed and 98^{th} percentile of reconstruction error is selected as the threshold value for anomaly detection. Threshold value is a tunable parameter and 98^{th} percentile of reconstruction error is selected based on trial and error on validation dataset. Trained autoencoder model and calculated threshold value is then used to detect anomalies in testing phase.

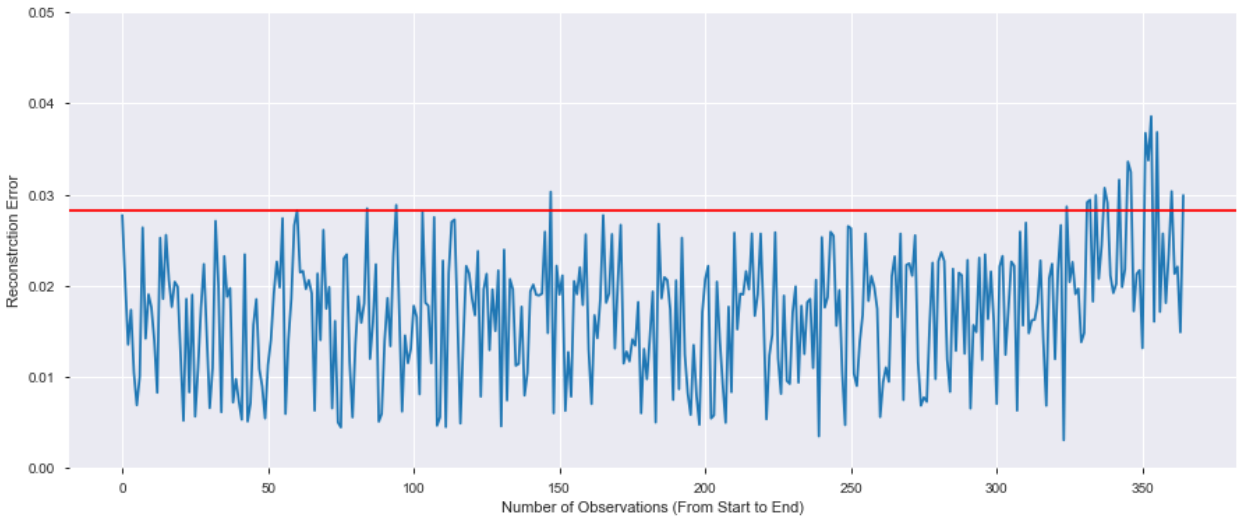
V. RESULTS

Performance of trained model is evaluated on test dataset consisting of run-to-failure data of 19 turbofan engines. As anomaly detection problem can be framed as a classification problem, therefore performance evaluation metrics used in this research are F1-score, Precision and Recall. For computing these metrics, true labels for each testing sample are required. This is achieved by assigning label 'Normal' to first 60% data and label 'Anomalous' to last 5% data of all testing (19) datasets. Reconstruction errors on two randomly chosen test examples for both the approaches are shown in Fig. 2 and Fig. 3. It is evident from both the Fig. 2 and Fig. 3 that reconstruction error increases as engine approaches failure (for both approaches). Threshold value computed by selecting 98^{th} percentile of reconstruction error on training set is also shown in the form of red horizontal line in following figures.

Results in Fig. 2 and Fig. 3 have shown that the best performance is achieved when no feature removal is applied. These results are also verified by F1-score computed on the test dataset for both the approaches. F1-score of approach without feature removal is 0.892 and for approach with redundant feature removal, F1-score is 0.813. These results are also presented in Table VI.

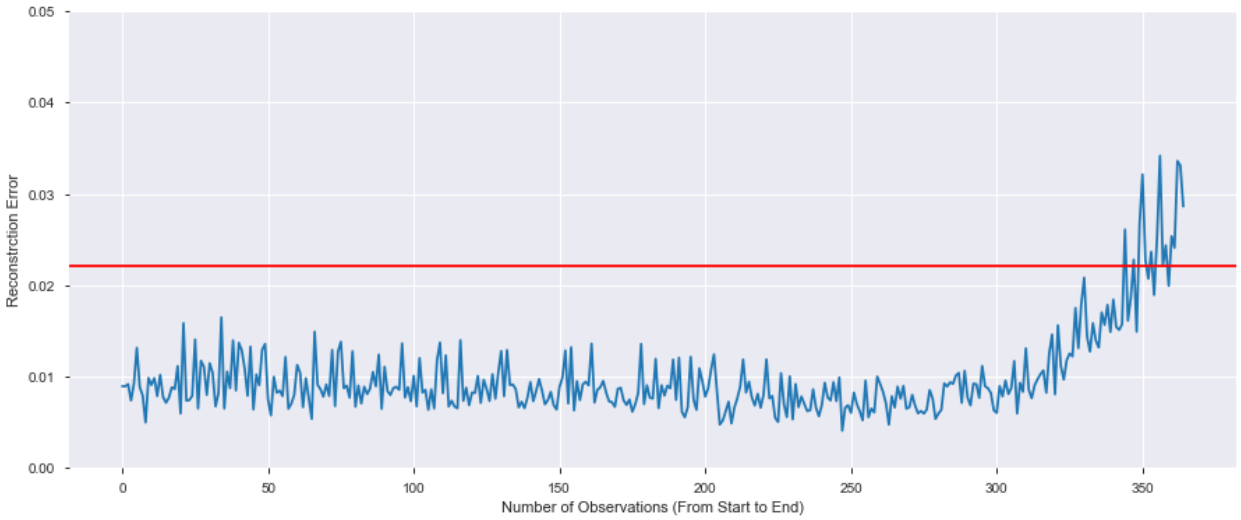


(a) Evolution of Reconstruction Error (without Redundant Features Removal).

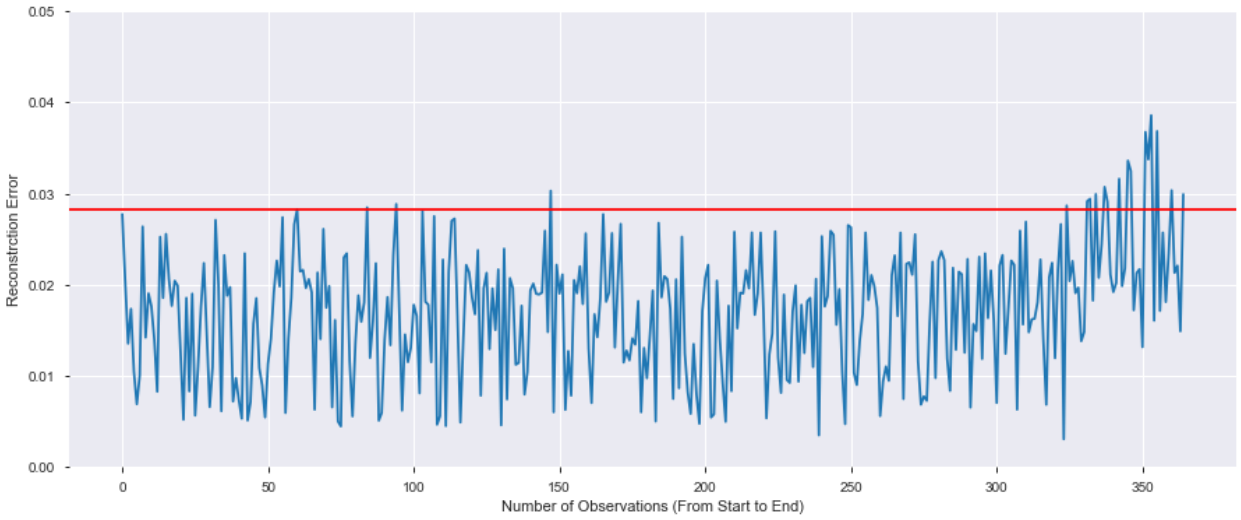


(b) Evolution of Reconstruction Error (with Redundant Features Removal)

Fig. 2. Evolution of Reconstruction Error for Engine # 235.



(a) Evolution of Reconstruction Error (without Redundant Features Removal)



(b) Evolution of Reconstruction Error (with Redundant Features Removal)

Fig. 3. Evolution of Reconstruction Error for Engine # 239.

TABLE VI. MODEL PERFORMANCE

	With Redundant Feature Removal	Without Redundant Feature Removal
F1-Score	0.813	0.892
Precision	0.704	0.896
Recall	0.611	0.724

VI. CONCLUSION

This paper proposed a semi-supervised autoencoder based anomaly detection approach to detect anomalies in turbofan engines. In the training phase, the autoencoder model is trained on data representing the normal behavior of turbofan engines. For tuning the architecture and hyperparameters of the autoencoder model, a Bayesian optimizing based approach was used. To study the effect of redundant features removal, two approaches are implemented and tested: with and without redundant features removal. For the removal of redundant features, Pearson's correlation was used to find a correlated set of features and one feature per set was used in training and testing. Performance evaluation metrics used in this research are F1-score, precision, and recall. Results have shown that the best performance is achieved when no redundant feature removal is applied. Our proposed approach has achieved F1score of 0.892, precision of 0.896 and recall of 0.724. This performance shows that autoencoders with optimal architecture can be a useful algorithm for the detection of anomalies in several real-world systems.

REFERENCES

[1] R. Mohammadi, E. Naderi, K. Khorasani, and S. Hashtrudi-Zad, "Fault diagnosis of gas turbine engines by using dynamic neural networks," in Turbo Expo: Power for Land, Sea, and Air, vol. 43987, 2010, pp. 365–376.

[2] R. Kandhari, V. Chandola, A. Banerjee, V. Kumar, and R. Kandhari, "Anomaly detection," ACM Comput. Surv, vol. 41, no. 3, pp. 1–6, 2009.

[3] P. G. Bringas and I. Santos, "Bayesian networks for network intrusion detection," Bayessian Network, editace A. Rebai, InTech, pp. 229–244, 2010.

[4] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representation by error propagation, parallel distributed processing," MIT Press, Cambridge, 1986.

[5] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems," Journal of network and computer applications, vol. 30, no. 1, pp. 114–132, 2007.

[6] T.-J. Zhou, Y. Li, and J. Li, "Research on intrusion detection of svm based on pso," in 2009 International Conference on Machine Learning and Cybernetics, vol. 2. IEEE, 2009, pp. 1205–1209.

[7] S. Omar and A. Ngadi, "H. jebur, h.(2013)," Machine Learning Techniques for Anomaly Detection: An Overview. International Journal of Computer Applications, vol. 79, no. 2, pp. 33–41.

[8] A. Fernandez, S. Garc´ ´ıa, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, "Foundations on imbalanced classification," in Learning from Imbalanced Data Sets. Springer, 2018, pp. 19–46.

[9] B. Lindemann, F. Fesenmayr, N. Jazdi, and M. Weyrich, "Anomaly detection in discrete manufacturing using self-learning approaches," Procedia CIRP, vol. 79, pp. 313–318, 2019.

[10] J. Dunn, "A graph theoretic analysis of pattern classification via tamura's fuzzy relation," IEEE Transactions on Systems, Man, and Cybernetics, no. 3, pp. 310–313, 1974.

[11] C. Liu, "Maximum likelihood estimation from incomplete data via emtype algorithms," in Advanced Medical Statistics. World Scientific, 2003, pp. 1051–1071.

[12] I. Syarif, A. Prugel-Bennett, and G. Wills, "Unsupervised clustering approach for network anomaly detection," in International conference on networked digital technologies. Springer, 2012, pp. 135–145.

[13] R. Fujimaki, T. Yairi, and K. Machida, "An approach to spacecraft anomaly detection problem using kernel feature space," in Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, 2005, pp. 401–410.

[14] P. Montague and J. Kim, "An efficient semi-supervised svm for anomaly detection," in 2017 International Joint Conference on Neural Networks (IJCNN). IEEE, 2017, pp. 2843–2850.

[15] F. J. Huang and Y. LeCun, "Large-scale learning with svm and convolutional for generic object categorization," in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1. IEEE, 2006, pp. 284–291.

[16] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, 2014, pp. 4–11.

[17] A. R. Triki, R. Aljundi, M. B. Blaschko, and T. Tuytelaars, "Encoder based lifelong learning," in ICCV, 2017.

[18] A. Al Bataineh and D. Kaur, "Optimal convolutional neural network architecture design using clonal selection algorithm," International Journal of Machine Learning and Computing, vol. 9, no. 6, 2019.

[19] A. S. A. Bataineh, "A gradient boosting regression based approach for energy consumption prediction in buildings," Advances in Energy Research, vol. 6, no. 2, pp. 91–101, 2019.

[20] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt: a python library for model selection and hyperparameter optimization," Computational Science & Discovery, vol. 8, no. 1, p. 014008, 2015.

[21] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in 2008 international conference on prognostics and health management. IEEE, 2008, pp. 1–9.

[22] A. Al Bataineh and D. Kaur, "A comparative study of different curve fitting algorithms in artificial neural network using housing dataset," in NAECON 2018-IEEE National Aerospace and Electronics Conference .IEEE, 2018, pp. 174–178.

[23] A. Al Bataineh, "A comparative analysis of nonlinear machine learning algorithms for breast cancer detection," International Journal of Machine Learning and Computing, vol. 9, no. 3, pp. 248–254, 2019.