ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

Development of an Aircraft Engine Health Monitoring System Using DBNN (Deep Belief Neural Network)

Samuel Chiwoke Amadi[1*]; Prema Kirubakaran[2]; Ridwan Kolapo[3]

[1,2,3]Department of Information Technology and Information Systems, Faculty of Computing, Nile University of Nigeria, Abuja

Corresponding Author: Samuel Chiwoke Amadi[1*]

Publication Date: 2026/11/05

Abstract: The fact that aviation could remain reliable, remain efficient and cost-saving is closely related to aircraft engine reliability yet the outdated system of relying on a particular set of thresholds and choosing a time to conduct a check-up is no longer applicable. There is the likelihood of the new aircrafts crashing in a mess and non-detection by simple rules which would have identified the aircraft as the right one. This was what led us to the creation of a brilliant health monitoring system, the system was developed on the basis of a deep belief neural network. It is fundamentally structured to uncover the faults at an early stage and will do the maintenance requirement prediction in a more superior manner. The network was trained to retrieve the crucial information of all types of sensors placed in the aircraft such as temperature, pressure, vibration and speed and all the others. The network has been constructed utilizing the unsupervised learning utilizing the restricted Boltzmann machines, the network has been refined utilizing the fine tuning of the model utilizing the restricted Boltzmann machines. The network has been in a position to detect the wear on the engine and the abnormal patterns in the data and has also been able to work with the noisy data. The network can forecast the sickness and engine issues before it is too late and it notifies the engine issues that are peculiar and peculiar patterns. In this manner, it can save a significant amount of time, resources, maintenance costs and save a significant amount of money. Here we also got the improvement in the performance of the neural network over the traditional machine learning models and through this we are able to see the performance improvement.

Keywords: *Aircraft Engine Health Monitoring; Predictive Maintenance; Deep Belief Neural Network; Restricted Boltzmann Machine; XGBoost; Fault Classification.*

How to Cite: Samuel Chiwoke Amadi; Prema Kirubakaran; Ridwan Kolapo (2026) Development of an Aircraft Engine Health Monitoring System Using DBNN (Deep Belief Neural Network). *International Journal of Innovative Science and Research Technology*, 11(10), 2884-2891. https://doi.org/10.38124/ijisrt/26oct1371

I. INTRODUCTION

They have extremely robust safety regulations in the aviation sector and have never left a single error behind, particularly the engines (Tahan et al., 2017). Engine performance does not only influence the efficiency inside the flight but the passenger's safety as well and leaves a significant effect to the economic situation of the aviation industry too. A significant component that will get out of hand in the event of hitches, fatalities, astronomies expenses and a sour reputation of the organization. The engines used to be inspected when a failure occurred which was in the old times. Then in due course, airlines have resolved to initiate recurrent checkups following a specified number of flights but this was insufficient. The companies had to spend a lot of money due to the need to replace parts on a case basis that was not that urgent as it should have been (Lee et al., 2014). Today, the engine provides live data to the crew members, and the smart tracking instruments are self-reliant They constantly examination of the engines and also record such data as the exhaust temperatures, the extent of vibration of the engine, the oil pressure and the velocity of the fuel flow. Several attempts at applying the antiquated machine learning practices (Support vector machines, Random forests or shallow neural nets) to identify and label failures have been made (Eker et al., 2012) but the issue with these methods is that they frequently require experts to create features and, in many cases, they fail when the intricacy of the issue is elevated (Janssens et al., 2016).

Deep learning, however, is an alternative method. In this step of selecting, Deep learning systems automatically consider the most important variables without human participation in selecting the important variables. The raw

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

data entering to these systems is usually composed of data that has been sent by sensors. As an example, a convolution neural network (CNN) is a popular architecture of deep learning in aero-engines. The latter is explained by the fact that CNNs can well process vibration patterns and may be employed in detecting faults/failures (Zhang et al., 2017). RNNs or LSTMs may frequently be utilized to represent patterns in the development of engine readings through time (Yuan et al., 2016). However, there are other problems. To begin with, deep networks are expensive to train with many examples tagged and, in this business, these examples are hard to obtain because of the poor failure rate of engines. Moreover, engineers are not usually aware of how such networks operate, and they usually break down in cases that are hard to identified. A case in point is Deep Belief Networks. DBNs are a form of deep learning architecture constructed with sub- pile mounted Boltzmann machines or RBMs. Their description was initially made by the team of Hinton in 2006. The characteristic of DBNs that sets them apart with the other types of deep learning architecture is that they are a mix between first learning the patterns between the aircraft behavior in an unsupervised mode with unlabeled data and then refining those patterns to the specific task using supervised feedback. The hybrid model is also suitable in the aviation sector since very few facts about actual faults or failures are known (Tamilselvan, V. S., and Wang, P., 2013). Instead of making an a priori assumption of how these faults or failures will occur, DBNs treat as input huge amounts of flight logs, the ones not discarded by carriers because of the possibility of fault/failure diagnosis in the future, to find normal aircraft patterns. After which, having a limited number of tagged examples, they optimize such patterns to pick faults and to pick up situations that were never anticipated before so that they can yield more out of airline data that will be of better use since real malfunctions are very rare.

The scientists realised that DBNs were very successful in following the state of a machine as well as being successful in recognizing the state of failure in a machine. The first applicants of DBNs to sensor-heavy diagnostics of jet engines are Tamilselvan and Wang in 2013. Their model was superior as compared to generic models in the NASA engine dataset. Second were those of Guo in 2016 where they set up a DBN model to find faults in bearing. Their version worked well in deriving significant patterns in messy vibration signals.

Relying on maintenance based on schedule is not adequate to the modern advanced engines. They produce excessive sensors that communicate excessive real time data. Virtually nothing is paid attention to, it is squandered instead of the fact that it could predict improved predictions (Zhao et al., 2019). Damage to the engine is not black and white. Instead, you are left with compound issues in which one aspect propels another and in the long run, minor faults develop. You require intelligent systems that are capable of identifying concealed hints in the complex signals to incorporate trouble or in anticipation (Tamilselvan and Wang, 2013). Fixing airplanes when they crash is the same process whereby sudden failure takes place, flights become late and passengers are put at risk. What you required are robust forecasting mechanisms that will be able to identify glitches in-time to ensure that planes remain safe and take off (Ellefsen et al., 2019).

Most of the architectures of the monitoring are grounded on the most basic models of Machine learning (support vectors machines, random forests, or shallow neural networks) (Zhao et al., 2019; Wang et al., 2020). Though these models might work well in good and predictable conditions these models will get out of order very quickly when you supply live sensor measurements. It is then succeeded by the necessity to inject havoc into features that have been found by experts who end up developing features by hand to find meaning (Tamilselvan & Wang, 2013). It is tiresome, time-consuming work. To overcome this, some scholars had to use more powerful tools such as CNNs or LSTyms. These are effective in trend location over time - which is optimum, except they need a mass of tagged instances. Such information does not lend itself to air transportation because engine failures which translate to a failed engine are rather rare, not to mention locating clean labels, which are very hard to find (Zack et al., 2018). Moreover, many of the arrangements fail to operate in case of garbled sensor readings in actual flights. In this regard, they overlook minor clues that allow to estimate the existence of an engine (Lee et al., 2018). What's not here yet? A learner system, independent in itself - including naked signals, discovering hidden regularities by itself but giving explicit and reliable answers to varying immediate surroundings.

We are developing a Deep Belief Neural Network that will be used to check jet engines. Zero tags were required in the early phases of learning and in the course of learning, the model learns through stacking up layers with assistance of Restricted Boltzmann Machines. This way it can work off raw sensor values, recognize significant patterns as well as what normal behavior comprises but does not need to search through hundreds and thousands of tagged breakdown cases.

II. RELATED WORK

- *Approach*

At this point, the majority of monitoring systems adhere to treat simple machine learning: support vector machines, random forests, shallow neural nets (Zhao et al., 2019; Wang et al., 2020). Of course, it works when it is clean and predictable. They can however not cope once you start passing along live sensor data to them. Instead, one day we are forced to reach out to specialists who will have to dig through sloppy data manually and create features to extract any information (Tamilselvan & Wang, 2023). It's slow, it's exhausting.

To bypass this, other authors adopted the use of powerful methods such as CNNs. They do patterns well but require hundreds of categorized data. That is a challenge in aviation, where engine failures are uncommon, and well-illustrated case studies are hard to locate (Zack et al., 2018). Besides, many of these systems fail when sensor data in the real world varies. They also do not notice minor indicators that might alert about the remaining life of an engine (Lee et al., 2018).

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

So, what's the answer? Expand self-teaching systems of building. Allow them to excavate raw signals, find patterns themselves, and provide clear and trustworthy answers even as all the things around them continue to evolve.

➢ *Research Gaps*

Looking at recent studies, it's easy to see major obstacles that slow down deep learning when tracking aircraft engine condition. Take the main issue being data. Cool supervised systems say, CNN-LSTM setups (Li et al., 2020) or focus-based LSTMs (Zhang et al., 2021) might score high in tests. Here is the problem though, tagged data is hard to come by, since aircraft hardly ever share detailed fault logs.

Other researchers have undergone other techniques. Even though the transfer learning (Cheng et al., 2021) can come in handy, the gap remains open. Contrasts-supervised on themselves (Wang et al., 2024)? Still playing catch-up. We are subtracting 20-30-percent compared with fully supervised systems.

The issue of interpretation and cognition of the end results remains. Many engineers apply post-factum techniques of analysis, including SHAP or LIME (Rajesh et al., 2022). These are effective, and yet consume time, and are costly to maintain. What is even more disturbing about it is the fact that they do not feature in the model, to start with but they are added afterwards and they tend to provide contradictory explanations.

Studies vary widely in how to respond to multiple time levels of depreciation modeling or how to come up with inputs of different sensors. As an example, multi-level CNNs (Zhao et al., 2022) are effective at the identification of trends at different timing layers. They however, intensely rely on the selection of ideal scales by specialists on readjusting setting repeatedly. Those models lack the tools to identify the most appropriate time related patterns by themselves. Neural networks (Ren et al., 2020) are good at following spatial positions but they lose strength once temporal order becomes key.

Most deep learning systems pay attention to just one job at a time, using examples with labels or tags. They put existing patterns in the same group, classifying them together instead of creating new patterns. Because of that, they end up underestimating important factors, understanding how engine data actually spreads out, catching odd outliers by noticing mismatches, or working well with mixed batches of tagged and raw info when labels are scarce.

What's lacking? A model that deals with low-data conditions by depending on self-supervised learning upfront, makes understanding easier by using gradual and continuous visual feedback on features, includes uncertainty tracking right from day one, detects complicated changes across different time spans, gathers diverse sensor inputs irrespective of the type and is still fast enough for live operation.

III. METHODOLOGY

➢ *Data Acquisition and Preprocessing*

The dataset in itself is a publicly available one dedicated to predictive maintenance with a high concentration of operational sensors data in aircraft engines. 10,000 data records and 10 attributes and various target variables associated with engine health are mixed with both numerical and categorical values. The target variable itself is not a yes-or-no, a few types of failure, such as heat dissipation, power, wear on the tool, overstrain, and normal working type fall under the target variable and this is a multiclass classification problem.

- *Data Loading and Exploration*

Using the Panda library, we loaded the data into Python to provide some initial exploratory analysis to familiarize ourselves with feature distributions, type of data, class balances and data shape. This provided us with a clear picture of what we were dealing with completing the construction of any models.

- *Handling Missing and Duplicate Data*

Extensive validation revealed that the dataset was clean without missing values, or duplicates or anything to repair or complete. We therefore put everything there intact and ensuring that the information remained intact and credible.

- *Normalization and Encoding*

The first step was to bring the data to the same scale all the numerical data, such as air temperature, process temperature, rotational speed, torque, and tool wear. By doing so, a feature does not stick out due to its higher numbers. On the categorical stuff, such as product type or failure type, I would convert them to numbers so that the standard machine learning models could model them can actually use them, as well as the deep learning benchmark.

➢ *Feature Selection*

Then we concentrated on putting in the functionality that is actually useful in forecasting engine failures and made things go fast with the classical models. I began examining the relationship between each feature and the target, i.e. ensuring there are no features that do not contribute much, or simply recount what we already know. The next part was to use Random Forest to rank each feature based on the extent to which it is useful in classification. Such characteristics as rotational speed, torque, tool wear, and the temperature related ones continued to stay on top. This two-step method simplified and streamlined the work of the classic models and also provided a good base in the comparison of the models with the DBN.

➢ *Feature Transformation and Preparation*

After getting the right features we narrowed it down to maximize each model. To maintain the distances and gradients-sensitive models, I re-scaled the numbers. All of this was changed into a format that could be read by the models (be they fixed-feature classical models or deep learning models that learn their own features). This maintained the

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

information congruent and leaving the DBN to its business and digging deeper to uncover the concealed trends.

➢ *Principal Component Analysis (PCA)*

Principal Component Analysis (PCA) was used to reduce redundancy and make things clean. Under PCA, I reduced the number of features and converted readings, which are correlated among the sensors, such as temperature, rotational speed, torque, and tool wear to independent components. This assists in bringing out the actual engine operation and eliminates the noise. In the classical models, PCA stabilizes training and increases the generalization of the models. In the case of the DBN, it provides small provision of information-filled inputs that allows the lower levels to concentrate on what is of actual interest than becoming frustrated with noisy inputs.

➢ *Data Partitioning*

As an evaluation method, we employed cross-validation in ensuring reliability of results. I stratified the data in such a way that each category of failures was fairly represented in each fold, which is very important in such a multiclass predictive maintenance exercise. Instead of 70/30, which was used in the classic models, I submitted the DBN to cross-validation, as deep learning is recommended to do, and also to ensure that the findings would stand.

➢ *Development of the Predictive Maintenance Models*

This section involves applying the actual predictive models and compare the traditional machine learning methods with deep learning in predicting engine failures.

- *Classical Machine Learning Models*

I tested and experimented with four supervised classical machine learning algorithms:

✓ K-Nearest Neighbors (KNN): It considers the proximity to things in the feature space.

✓ Support Vector Machine (SVM): It constructs optimum boundary between classes.

✓ Random Forest Classifier (RFC): It is a randomly chosen set of decision trees and it is enhanced to improve the accuracy and minimized variance.

✓ Extreme Gradient Boosting (XGBoost): It is a boosting algorithm which corrects the mistakes in an intelligent, step-by-step process.

Each of these models is based on hand-picked and machinery-engineered features and we set their hyperparameters so that everyone had an equal opportunity.

- *Deep Belief Network (DBN) Implementation*

On the deep learning benchmark, I selected a Deep Belief Network (DBN). The model positions Restricted Boltzmann Machines (RBMs) on top of each other in my model, and allows it to learn more and more abstract data representations at every level. This is an attractive aspect of the DBN as in contrast to the classical models, it does not need supervision to learn its own characteristics. It is in a position to find concealed relations and patterns within the data that we have not explicitly ought to be coded in.

The training of the DBN occurs in two phases. It then, does unsupervised pretraining using RBMs in order to determine the inner structure of the data. Subsequently, it is followed by fine-tuning that is supervised to identify the exact failure categories. Such a bi-step procedure renders the DBN a formidable challenge on the classical models.

IV. RESULTS AND DISCUSSION

The following tables reproduce the performance values reported in the thesis. Values are shown to three decimal places to preserve the reported precision. They are not recalculated from raw predictions because the prediction files and confusion-matrix counts are not reproduced in the manuscript source used for this revision.

Table 1 Validation Performance of Classical Models

Metric	KNN	SVC	RFC	XGBoost
Accuracy	0.956	0.968	0.975	0.981
AUC	0.956	0.993	0.998	0.999
F1-score	0.957	0.969	0.975	0.981
F2-score	0.957	0.968	0.975	0.981

The validation results show strong reported performance for all four classical classifiers. XGBoost achieved the highest validation accuracy (0.981) and AUC (0.999), with Random Forest next at 0.975 accuracy and 0.998 AUC. SVC produced 0.968 accuracy and 0.993 AUC, while KNN produced 0.956 accuracy and 0.956 AUC. These values suggest that the tree-ensemble methods performed particularly well on the structured features used in the experiment. They do not, on their own, establish generalization to engines, operating conditions, or failure types not represented in the evaluation data.

Table 2 Test Performance of Classical Models

Metric	KNN	SVC	RFC	XGBoost
Accuracy	0.966	0.973	0.972	0.985
AUC	0.956	0.995	0.997	0.999
F1-score	0.966	0.973	0.972	0.985
F2-score	0.966	0.973	0.972	0.985

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

On the reported test results, XGBoost achieved the highest accuracy at 0.985 and an AUC of 0.999. SVC reached 0.973 accuracy and 0.995 AUC, while RFC reached 0.972 accuracy and 0.997 AUC. KNN achieved 0.966 accuracy and 0.956 AUC. The difference between the validation and test values is small for the reported classifiers. However, a high score is meaningful only in relation to the target label, class balance, the data split, and possible overlap between related engine records. These elements should be confirmed before interpreting the scores as evidence of operational readiness.

Table 3 DBNN Performance Reported in the Thesis

Metric	Validation	Test
Accuracy	0.801	0.805
AUC	0.863	0.866
F1-score	0.729	0.738
F2-score	0.769	0.775

The DBNN achieved a reported test accuracy of 0.805, AUC of 0.866, F1-score of 0.738, and F2-score of 0.775. These values are lower than the corresponding reported scores for XGBoost. The results therefore do not support a claim that DBNN outperformed the conventional models in this experiment. The DBNN results may reflect model configuration, training-data size, feature representation, target complexity, or other implementation choices, but the available results do not isolate the contribution of any one factor. These explanations should be treated as possibilities for future testing rather than established causes.

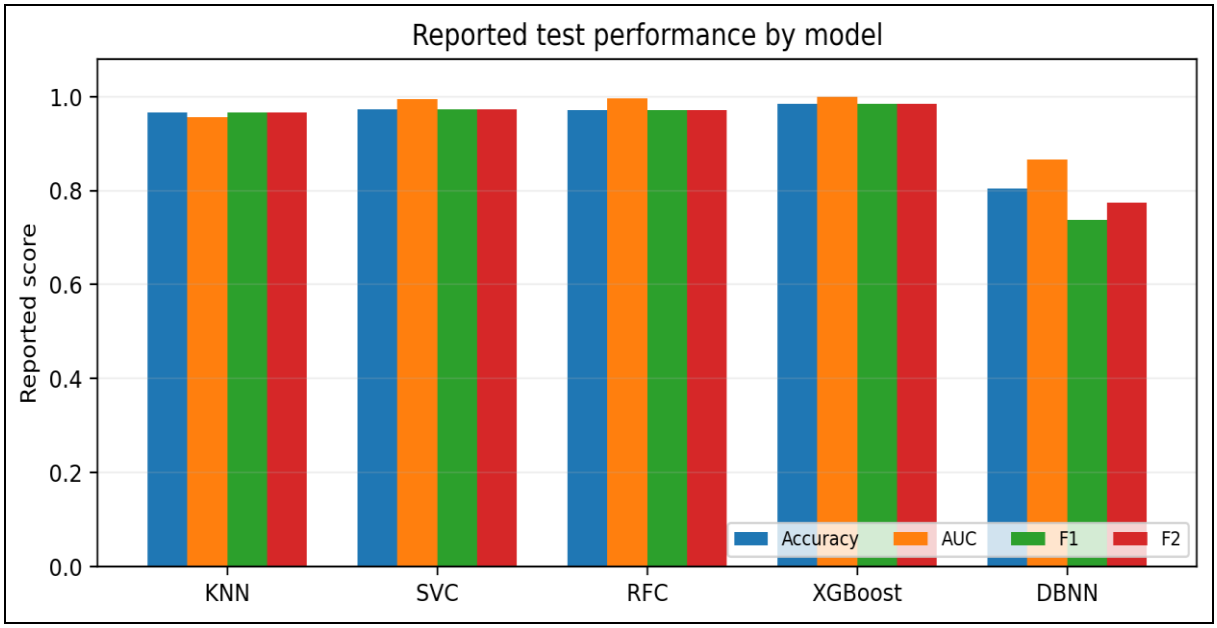


Fig 1 Reported test Accuracy for the Four Classical Models and DBNN.

➢ *Interpretation and Limitations*

The comparison indicates that a deep architecture is not automatically the strongest choice for structured tabular engine-monitoring data. XGBoost performed best on the reported test accuracy and AUC, whereas DBNN recorded lower values across the listed metrics. This is a useful empirical finding: method selection should be guided by validated performance and the intended use case, not by model complexity alone. In practice, recall, false-negative rates, calibration, and the cost of missed faults would also be important to assess before using a model to support maintenance decisions.

There are two important limitations to the present comparison. First, the thesis describes a 70/30 split for the classical models and cross-validation for DBNN. Unless the same test cases and evaluation procedure were used for all models, the scores should not be treated as a perfectly controlled head-to-head comparison. Second, the dataset description combines a reference to NASA C-MAPSS with a statement that the working dataset contains 10,000 records and 10 attributes. The exact subset and preprocessing output need to be documented. Other information needed for reproduction includes class distribution, model hyper parameters, random seeds, confusion matrices, and per-class precision and recall. The results are reported as supplied by the project and should be checked against raw outputs before journal submission.

Objectives 1 and 2 focused on the design and implementation of a proper predictive maintenance system that was able to categorically label various engine failures. We achieved these targets through four classic model machine learning models, namely, K-Nearest Neighbors (KNN), Support Vector Classifier (SVC), Random Forest Classifier (RFC), and Extreme Gradient Boosting (XGBoost). Another

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

deep learning benchmark is hereby a Deep Belief Network (DBN) that we brought in.

In the case of the classical models, we trained and evaluated them, using a stratified train-test split (70/30), ensuring each failure mode was well-represented. Our features of interest were selected using correlation and Random Forest features importance, and Principal Component Analysis (PCA) was then conducted to select only the most useful features, without redundancy in their representations.

The DBN was treated in a different way, cross-validation, which is typical of deep learning, was used to obtain a more difficult and realistic measure of its generalization performance. The DBN uses learned complex, hidden features instead of hand-picked ones, achieving sub-optimal pretraining. This allows it to identify more profound patterns of failure which would be overlooked by more traditional models.

Our measures of model performance were standard multiclass classification measures: Accuracy (ACC), Area Under the Curve (AUC), F1-score, and F2-score. These provided us with a decent balance between the overall performance and the capability to detect failures, which is highly essential in the field of aviation.

All the classical models have high predictive power as viewed by the validation results. Ensemble techniques obviously dominated the stage with the XGBoost winning the race by a margin of 98.1% validation and 0.999 AUC. Random Forest followed closely, which was again a testimony to how ensemble learning can address the non-linear engine data. KNN was slower, that is, it is not unexpected that this method is sensitive to the geometry of the feature space, and the overlap of classes.

Similar story was told by test data. XGBoost remained at the top reaching 98.5% accuracy and AUC of 0.999 on unseen engine data. Random Forest and SVC were also doing pretty well as KNN was left behind.

Its glance at the DBN showed that it was doing a fine job, although failing to match the standards of the classical ensemble paradigms. Its unsupervised pretrain makes the DBN superb at learning non-linear and hierarchical features, however, its accuracy and F-scores continued to be lower than XGBoost and Random Forest. It was found that the DBNs are highly representative learners, but their output in this case was constrained by the size of the dataset, the structure of the feature, and computationalism. Therefore, deep learning does not necessarily outperform classical models with structured tabular data such as this one.

The confusion matrices provided us with a more detailed view of how each of the models reacted to the various types of engine failures. Ensemble models were hardly confused by categories, indicating that they are effective at distinguishing the failure modes. KNN, in turn, proved to be less favorable, particularly when it comes to different types of failures such as heat dissipation and power failure- perhaps, due to reliance on local data structure.

The feature distribution plots make this rather obvious-failure and non-failure states are distinctly clear on some important variables, in particular rotational speed, torque and tool wear. These are good predictors. There is much more overlap when you consider features related to temperature, however. They contribute, but they are not as contributing to tell the difference between the two states themselves.

As the visualization reveals, the models trained on the full and optimized set of features always win over the models trained on smaller subsets. Ensemble models have many advantages in terms of their stability and reliability--XGBoost, in this case, leads in all the indicators.

With the models, there is a definite pecking order when stacked. The classical ensemble models, and XGBoost, in particular, often beat standard classifiers as well as deep learning models in terms of accuracy, AUC, and F-score. The DBN is a good comparison to the state-of-the-art, though its results indicate that there is something worth noting: deep learning models are excellent at discovering the concealed patterns, but it does not mean that they are always bound to provide the best predictions on ordered predictive maintenance data. Relative to their more intricate configurations, relatively simple ensemble models actually often perform better in practice, and with less computing power.

V. CONCLUSION AND RECOMMENDATIONS

This chapter combines the significant findings, draws conclusions, and provides some recommendations and practical tips on how to work in the future. The talk is a hybrid of both engineering and machine learning aspects, being about the relative capability of classical models versus a deep learning benchmark stack to be used in aviation predictive maintenance. Although it is aviation-related in this case, it can also be applied to other forms of safety-critical and industrial maintenance systems.

- *Summary of Major Findings*

In general, the predictive maintenance system implemented on the basis of DBNN training was trained and demonstrated good results without any issues. Such findings indicate that this method is effective and the simulated aircraft engine behavior is more realistic than the commonly employed machine learning models that we compared to. Although the DBN failed to do better than the ensemble models, the process was not in vain as the DBN served as a benchmark and provided us with significant information about the deep learning role in aviation maintenance. The system is a trustworthy, scalable, and comprehensible basis of Intelligent Predictive maintenance of aircraft engines all in all.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

➢ *Conclusion*

The study achieved its objectives through designing, construction and testing of a Deep Belief Neural Network (DBNN) model to identify engine failure of an aircraft. The research demonstrates that both the conventional machine learning methods and the state-of-art deep learning methods are capable of working with high-dimensional sensor measurements and predicting engine malfunctions. It is an engineering perspective that this system assists in facilitating proactive maintenance - identify issues at their beginning, reduce unplanned outages, and improve aviation safety. Although the DBN is an advanced approach based on deep learning, XGBoost, nevertheless, emerged victorious in practical use. These findings help one understand that the model of predictive maintenance that is best is not simply the complexity of the algorithm. It is being all about fitting the correct data, characteristics, and learning approach. Despite these obstacles, such as data and system integration, the study provides a robust foundation on smart maintenance systems in the aviation industry and other areas that are related to high safety standards.

➢ *Recommendations*

Consequent to these findings, the following may assist in increasing the actual performance and implementation;

- Incorporate the use of automated predictive maintenance systems, into the existing aircraft maintenance, health monitoring, and decision-making systems to enable the possibility of constant, real-time failure detection.
- Use deep learning models such as DBNs as references and to experiment with more complicated scenarios, in particular with large data sets or more complex time-based features.
- Include human in the loop capabilities: in this way the maintenance engineers can preview and validate the system predictions before any action is undertaken.
- Establish sound data governance and security to include reliability of sensor, data integrity, and cybersecurity as well as compliance with regulations.
- Explainability tools and visualization tools must be embedded into the system. The only solution to maintain transparency and make people trust the result is that last though with deep learning or ensemble models that might get quite opaque.

➢ *Future Research Directions*

The next area of research should be as follows:

- Integrate classical models and deep learning - hybrid and ensemble classification models can combat adverse effects of traditional methods: clarity and trustworthiness offered by conventional models, and unraw power of deep learning.
- Introduce temporal and sequential modeling (b). Such methods as recurrent neural networks or time-series training can monitor the process of failures formation and assist in forecasting the remaining life span of a system.
- Work with larger, more diverse data. Incorporate data on other types of aircrafts and settings. It is then the only way of knowing whether these models actually work all over the board or not.
- Model explainability tools are likely used to interpret the outputs, such as: SHAP models Attention mechanisms
- This assists not only in the process of understanding, but also in fulfilling the regulatory requirements in aviation maintenance.
- Test these systems on real-time basis. Test their performance with real world problems: latency, noisy sensor streams, streaming data- does it handle the load?

REFERENCES

[1]. A. K. Jardine, D. Lin, and D. Banjevic, “A review on machinery diagnostics and prognostics implementing condition-based maintenance,” Mechanical Systems and Signal Processing, vol. 20, no. 7, pp. 1483–1510, 2006.

[2]. J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, “Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications,” Mechanical Systems and Signal Processing, vol. 42, nos. 1–2, pp. 314–334, 2014.

[3]. G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,” Neural Computation, vol. 18, no. 7, pp. 1527–1554, 2006.

[4]. A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, “Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture,” Reliability Engineering & System Safety, vol. 183, pp. 240–251, 2019.

[5]. O. Fink, Q. Wang, M. Svensén, P. Dersin, W. J. Lee, and M. Ducoffe, “Potential, challenges and future directions for deep learning in prognostics and health management applications,” Engineering Applications of Artificial Intelligence, vol. 92, article 103678, 2020.

[6]. M. Tahan, E. Tsoutsanis, M. Muhammad, and Z. A. Abdul Karim, “Performance-based health monitoring, diagnostics and prognostics for condition-based maintenance of gas turbines: A review,” Applied Energy, vol. 198, pp. 122–144, 2017.

[7]. A. Saxena and K. Goebel, “Turbofan engine degradation simulation data set,” NASA Ames Prognostics Data Repository, NASA Ames Research Center, 2008. [Verify dataset citation/version against the exact files used.]

[8]. M. Arias Chao, C. Kulkarni, K. Goebel, and O. Fink, “Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics,” Data, vol. 6, no. 1, 2021. [Verify volume/article details against the original publication.]

[9]. M. Baptista, E. M. Henriques, and I. P. de Medeiros, “Comparative study of deep learning models for remaining useful life estimation of aircraft turbofan engines,” Engineering Applications of Artificial Intelligence, 2022. [Verify full bibliographic details against the source cited in the thesis.]

[10]. R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, “Deep learning and its applications to machine health monitoring,” Mechanical Systems and Signal Processing, vol. 115, pp. 213–237, 2019.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26oct1371

[11]. W. Zhang, D. Yang, and H. Wang, “Data-driven methods for predictive maintenance of industrial equipment: A survey,” IEEE Systems Journal, vol. 13, no. 3, pp. 2213–2227, 2019.
[12]. A. K. S. Jardine, D. Lin, and D. Banjevic, “A review on machinery diagnostics and prognostics implementing condition-based maintenance,” Mechanical Systems and Signal Processing, vol. 20, no. 7, pp. 1483–1510, 2006. [Duplicate thematic source in the thesis bibliography; replace with the intended distinct source if applicable.]