

University of South Alabama

JagWorks@USA

---

Undergraduate Theses

Honors College

---

2021

## A Hybrid Decision Tree - Neural Network (DT-NN) Model for Predictive Maintenance Applications in Aircraft

Jarrold Carson

*University of South Alabama*

Follow this and additional works at: [https://jagworks.southalabama.edu/honors\\_college\\_theses](https://jagworks.southalabama.edu/honors_college_theses)



Part of the [Other Computer Sciences Commons](#)

---

### Recommended Citation

Carson, Jarrod, "A Hybrid Decision Tree - Neural Network (DT-NN) Model for Predictive Maintenance Applications in Aircraft" (2021). *Undergraduate Theses*. 11.

[https://jagworks.southalabama.edu/honors\\_college\\_theses/11](https://jagworks.southalabama.edu/honors_college_theses/11)

This Undergraduate Thesis is brought to you for free and open access by the Honors College at JagWorks@USA. It has been accepted for inclusion in Undergraduate Theses by an authorized administrator of JagWorks@USA. For more information, please contact [jherrmann@southalabama.edu](mailto:jherrmann@southalabama.edu).

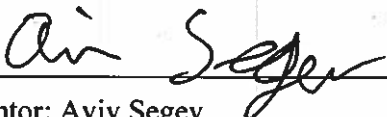
A Hybrid Decision Tree – Neural Network (DT – NN) Model for Predictive Maintenance  
Applications in Aircraft

By  
Jarrod Carson

A thesis submitted in partial fulfillment of the requirements of the Honors College at the  
University of South Alabama and the Bachelor of Science Computer Science in the Computer  
Science Department


University of South Alabama  
Mobile  
May 2021

Approved by:

  
Mentor: Aviv Segev

  
Committee Member: Tom Johnston

  
Committee Member: Ryan Benton

  
Kathy J. Cooke  
Dean, Honors College

© 2021  
Jarrod Carson  
ALL RIGHTS RESERVED

## **ACKNOWLEDGEMENTS**

I would like to thank my mentor, Dr. Aviv Segev, for introducing me to and guiding me along the path of research. If not for his guidance I may not have gained the same knowledge I have. This work has afforded me many opportunities to put myself out there professionally, to learn and share knowledge, and to develop my mind.

I would also like to thank the University of South Alabama Honors College for assisting me on this journey and providing me with experiences I won't forget.

Additionally, I would like to thank the friends I've made since attending the University of South Alabama for giving me lifelong connections I'll cherish until the end of time.

Finally, I would like to thank my family for encouraging me to keep going against all odds, and that I can achieve anything if I put my mind to it.

## **ABSTRACT**

As the Age of Information has evolved over the last several decades, the demand for technology which stores, analyzes, and utilizes data has increased substantially. For countless industries such as the medical, retail, and aircraft industries, such technology is crucial to their operation. This project proposes a hybrid machine learning model consisting of Decision Trees and Neural Networks which is able to classify data of varying volume and variety effectively and efficiently. The model's structure consists of a decision tree with each node of the tree containing a neural network trained to classify a specific category of the output using binary classification. To validate the model's efficacy, it is tested by applying it to a dataset consisting of the Federal Aviation Administration's (FAA's) Boeing 737 maintenance data, consisting of 137,236 unique records, each comprised of 72 variables, in a predictive maintenance setting. The predictive maintenance is performed by classifying the Discrepancy variable, a free-text descriptor of the maintenance issue faced by the aircraft, by first determining if the issue occurred during scheduled maintenance or not, and subsequently breaking down the nature of the incident into more specific categories. Results indicate that this hybrid model is able to classify incidents with high accuracy and precision. Additionally, the model is able to identify the most significant inputs involved in classification allowing for increased model performance. This both demonstrates the model's applicability to real-world scenarios and showcases the benefits of combining Decision Trees and Neural Networks in a hybrid structure rather than using them individually.

## **TABLE OF CONTENTS**

ACKNOWLEDGEMENTS .....	iii
ABSTRACT.....	iv
LIST OF ABBREVIATIONS.....	vi
LIST OF FIGURES .....	vii
LIST OF TABLES .....	viii
INTRODUCTION .....	1
RELATED WORK.....	4
Aircraft Maintenance .....	4
Machine Learning, Ensemble Learning, & Hybrid Models .....	7
Decision Trees, Neural Networks, & Model Optimization .....	9
METHODOLOGY .....	11
Uniqueness of Problem.....	11
Problem Setting.....	11
Integrating Decision Trees and Neural Networks.....	14
Model Outline .....	16
Data Preprocessing.....	16
EXPERIMENTS & RESULTS.....	19
Data .....	19
Methods.....	21
Neural Network Pseudocode.....	22
Experiments .....	23
Results.....	26
Activation Function & Hidden Layer Tests.....	32
Pearson/Spearman Correlation Tests .....	33
Heatmap and Correlation Information .....	35
Adaptive Learning Rate .....	36
CONCLUSION.....	39
REFERENCES .....	40

## **LIST OF ABBREVIATIONS**

AI	–	Artificial Intelligence
ANN	–	Artificial Neural Network
AUC	–	Area Under the Curve
NB	–	Naïve-Bayes
DT	–	Decision Tree
FAA	–	Federal Aviation Administration
NN	–	Neural Network
ML	–	Machine Learning
PCA	–	Principal Component Analysis
RF	–	Random Forest
SL	–	Supervised Learning
SVM	–	Support Vector Machine

## **LIST OF FIGURES**

Fig. 1: Integration of Decision Tree – Neural Network.....	13
Fig. 2: Activation Function Used.....	21
Fig. 3: Decision Tree - Neural Network with Specific Classification Categories .....	24
Fig. 4: F1 Score vs. Number of Hidden Layers .....	25
Fig. 5: Accuracy vs. Number of Hidden Layers .....	25
Fig. 6: Comparing the Performance of Activation Functions.....	27
Fig. 7: Average Weights for Leading First Layer Inputs.....	27
Fig. 8: AUC, Accuracy, Recall, and Precision vs. Number of Inputs. ....	28
Fig. 9: F1 Score vs. Number of Inputs, Precision vs. Recall. ....	30
Fig. 10: Precision vs. Recall for Maintenance, Crack, and Fuselage Classifications. ....	31
Fig. 11: Precision vs. Recall Comparison Between DT - NN Model and NN. ....	31
Fig. 12: Pearson Correlation Heatmap for Crack.....	34
Fig. 13: Spearman Correlation Heatmap for Crack .....	34
Fig. 14: Pearson Correlation Bar Chart Between All Inputs and Discrepancy.....	38
Fig. 15: Spearman Correlation Bar Chart Between All Inputs and Discrepancy .....	38



## **LIST OF TABLES**

Table 1: Boeing 737 Dataset Composition .....	20
Table 2: Neural Network Pseudocode .....	22
Table 3: Pearson Correlation Between Discrepancy and Inputs.....	35
Table 4: Spearman Correlation Between Discrepancy and Inputs .....	35

## **INTRODUCTION**

When it comes to analyzing vast quantities of complex data it is beneficial to implement robust algorithms capable of processing, interpreting, and classifying the information effectively and efficiently. For many years, the field of Machine Learning (ML), a subfield of Artificial Intelligence (AI), has been invaluable in tackling these difficult tasks. However, with current data collection practices, the volume and complexity of data has increased tremendously causing ML to face new challenges when designing and implementing algorithms able to handle such data. To combat these challenges, interest in researching more robust algorithms has increased. For many industries ML is crucial and research into improving techniques is necessary [1] – [3].

One of the most common areas of ML is Supervised Learning (SL), the area of ML where data with labeled and known characteristics is used to classify or predict the outcome of similar data. This area consists of two primary stages: the training stage and the testing stage. During the training stage the data, consisting of a set of input and output variables, is fed to the ML algorithm for it to learn relationships between the input and output. During the testing stage the algorithm, given another dataset of input and output variables, attempts to predict the value of the output given the value(s) of the input by applying the learned relationships from the training stage. Many of the algorithms belonging to SL, such as Decision Trees (DT), Neural Networks (NN), and Support Vector Machines (SVM), are able to effectively perform these tasks on their own, however, they each possess their own strengths and limitations. For example: NN are excellent at classifying data of large volume although it is nearly impossible to understand the internal processes between providing input and receiving output. To alleviate some of the limitations of individual algorithms, hybrid models are devised.

Hybrid models consist of multiple ML models working in unison to solve problems. Recently, hybrid models have attracted attention due to their ability to augment the strengths and diminish the weaknesses of the individual models involved [4] – [7]. One such individual model is the Neural Network (NN) which is widely used in numerous fields as it performs very well when predicting for new data after being trained on existing data. Another common model is the Decision Tree (DT) which excels in classifying features of data. In this thesis, a hybrid model approach is proposed consisting of both DT and NN models. The structure of this model consists of a DT model where each node of the tree contains a specialized NN. These specialized NN are each catered towards different specific characteristics of the data, allowing for more effective classification results than if an individual model were used. Each node of the tree represents a categorical classification of the data with each child node representing a subcategory of the parent node which may further classify the data. This allows for data to be classified at differing levels of granularity with finer grain classification occurring near the tree's leaves. The hybrid approach proposed here utilizes a system of binary classification, classification using one of two possible categories, which structures the DT as a binary tree. The result is a binary tree where each node contains a NN specialized in performing binary classification on a specific category of the data where predictions become more detailed and specific as the tree is descended. Therefore, the height of the DT structure represents the quantity of details that the model predicts, with greater height values indicating greater detail.

The proposed hybrid model is applied to the field of predictive maintenance regarding aircraft. In predictive maintenance, a data-driven approach is taken to determine where and when a machine will fail. Typically, this is performed with ML where maintenance and operational data of the machine is analyzed to determine the conditions most likely to indicate mechanical failure.

For an aircraft, this data consists of real-time sensor data from many of the aircraft's components and subcomponents. The data used to train the proposed model is sourced from the Federal Aviation Administration's (FAA's) public Boeing 737 maintenance data which consists of 137,236 maintenance records each comprised of 72 unique variables. As this data is public and generated from usual operation of Boeing 737 aircraft rather than in a controlled environment it allows the proposed model to be applied to a real-world scenario using authentic operational data.

## **RELATED WORK**

### Aircraft Maintenance

According to operational regulations set forth by the FAA, aircraft are required to undergo periodic maintenance after so many hours of operation [8]. To minimize delays and better utilize aircraft such maintenance is performed at night, often requiring the aircraft to be stationed at a maintenance facility overnight every few days. Improper performance of maintenance may lead to the aircraft experiencing an accident [9, 10].

As experiencing an accident could cause catastrophic losses, both financial and of life, it is necessary for airline companies to have a plan to deal with crises caused by accidents. In such rapid response scenarios, an adaptive method for crisis ontology may be used to represent knowledge in these situations [11]. Using this method, the ontology of a crisis is extended to tailor it to the current crisis. Extending this method to identify appropriate humanitarian response in a crisis requires merging ontologies and logic rules to represent humanitarian needs [12]. Using the extended method is advantageous in identifying humanitarian needs and prioritizing responses to a crisis, allowing for decision makers to focus on implementing solutions without unnecessary attention being given to unrelated information.

Aircraft delays can be expensive for airliner companies, so proper scheduling is crucial for aircraft operation [13]. When forming the flight schedule for many aircraft, a problem arises in determining when aircraft should fly and when each aircraft should undergo the different levels of maintenance as set forth by the FAA. Therefore, scheduling is a delicate operation where care should be taken to minimize the costs of maintenance and aircraft reassignment.

In aircraft maintenance, an organization's safety management system is necessary for the monitoring and mitigation of safety risks. A self-regulatory model was developed to examine these safety management systems, focusing on the human and organizational characteristics, across different organizations involved in aircraft maintenance [14]. The model proved to be effective in analyzing relevant aspects of each organization's safety management system, however it was unable to adequately account for the significance of planning and change in these systems.

A study into the importance of situational awareness among aircraft maintenance teams was conducted [15]. Findings indicate that, in many environments, the situational awareness of the team members was critical for reducing errors and increasing performance. The findings also demonstrated that there existed barriers which inhibited situational awareness both between and withing aircraft maintenance teams.

To reduce the operational costs related to aircraft maintenance, a predictive line maintenance optimization was proposed [16]. In formulating the optimization problem, the optimization was subjected to multiple wear conditions. Degradation trends were extracted using the Kalman Filtering. The optimization involved minimizing operational costs according to numerous factors such as dispatch requirements, aircraft delay costs, flight cancellations, and equipment costs.

Models of cognitive error and distortion have been used to evaluate unsafe actions that resulted in accidents within safety-critical environments [17]. A majority of the models of accident causation are based on the idea that human error is among the contributing factors. However, at present there exists a lack of published information regarding connections between specific errors and their contributing factors. A survey reported that out of 619 safety incidents involving aircraft maintenance, 96% involved the actions of maintenance crew members [17]. Results specified

which types of errors were associated with specific sets of actions and the outcomes of such occurrences. Among the associations discovered are links between memory lapses and fatigue and between rule violations and time pressure from deadlines.

A short-term planning methodology for line maintenance activities of airline operators during an aircraft's turnaround time was proposed which provided decision making for postponing maintenance actions which impacted aircraft deployment with high fleet operability and low maintenance costs as the goals of this methodology [18]. The method involved a multi-criteria mechanism which evaluated a generated set of maintenance plan alternatives on the grounds of information pertaining to operational and financial constraints. Such alternatives were defined as the potential allocation of airport resources to all deferred maintenance activities. The criteria involved in the decision making were cost, remaining useful life of the plan, operational risks, and flight delay.

Recent research into the causes of aviation accidents demonstrates that increased air-transportation safety requires a reduction in the impact of human error on operations [19]. Due to stringent schedules for aircraft operation and time constraints, aircraft maintenance workers are often under large amounts of stress which contributes to human error. Using computer-based support systems errors may be mitigated by assisting aircraft technicians in understanding information related to their work. Computer systems can store and recall this information to minimize the errors from procedure violations, misinterpretation of facts, or insufficient training. At present, many factors such as unwieldly hardware, installing markers on aircraft, and the need to quickly create digital content appear to inhibit the ability of maintenance workers to perform effective maintenance of aircraft in industry.

## Machine Learning, Ensemble Learning, & Hybrid Models

The concept of ML was born of the study of the brain and how certain external events caused neurons to fire [20]. Ensemble learning is the concept of applying multiple ML models to collectively solve a problem. This differs from hybrid models in that the individual models within the greater ensemble model are independent. In ensemble learning each of these individual models is independently trained and tested on data before the models vote on the predicted outcome. In hybrid models the individual models work dependent on each other as one collective unit with the voting system absent.

There exist numerous examples of both ensemble learning and hybrid ML models. A proposed hybrid model, consisting of the maximum entropy model, SVM, and Naïve-Bayes (NB) was created for multi-document text summarization [21]. To improve the model's classification accuracy another hybrid model was proposed for multi-class problems [22]. This model consists of the C4.5 DT classifier and the One-vs-All approach, an approach to multi-class classification in which N-binary classification models are generated for N-classes of the data. The efficacy of the hybrid model was demonstrated when applied to open-source image segmentation, dermatology, and lymphography datasets. Another recently developed hybrid model which involved DT, Random Forest (RF), and Gradient Boosting was applied to water quality prediction [23]. The method, known as Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN), had two variants. The first is based on Gradient Boosting (CEEMDAN – XGBoost) while the second is based on RF (CEEMDAN – RF). Each of these variants of the method resulted in high performance when predicting different water quality parameters. A hybrid model consisting of NN and genetic algorithms was created and applied to a heart disease dataset to detect the presence of heart disease [24]. The combination of NN and genetic algorithms supplemented



the NN model and allowed for increased performance when detecting heart disease. An ensemble method constructed to assess the susceptibility of an area for a landslide consisted of numerous ML methods [25]. The method consisted of a type of NN known as a Multilayer Perceptron (MLP) which was integrated with AdaBoost, Bagging, Dagging, MultiBoost, Rotation Forest, and Random Subspace. A hybrid method designed to diagnose Type 2 Diabetes consists of K-means Clustering and the J48 DT [26]. To address condition-based maintenance in manufacturing industries, MapReduce and numerous classifier DT models with dynamic weight adjustment were used on data collected from maintenance activities [27]. A just-in-time prediction method was proposed utilizing ensemble learning [28]. An advantage of this ensemble model is that it effectively handles redundancy and imbalance within the data while maintaining a robust algorithm. A similar study into just-in-time prediction using ensemble learning methods was conducted which proposed a two-layer ensemble learning (TLEL) approach based on DT [29]. The model's outer layer uses multiple different RF models for training while the inner layer is a hybrid of DT and bagging methods to construct a RF model. A recent study utilized ensemble learning methods to increase predictive performance in determining the remaining useful life (RUL) of aircraft engines [30]. The approach involved numerous methods including RF, Classification and Regressing Tree (CART), Recurrent Neural Network (RNN), Autoregressive (AR) model, Adaptive Network-based Fuzzy Inference System (ANFIS), Relevance Vector Machine, and Elastic Net (EN). To achieve the best combination of weights the method used Particle Swarm Optimization (PSO) and Sequential Quadratic Programming (SQP). An extension of this approach utilized Directed Acyclic Graph (DAG) combined with Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) in predicting RUL [31]. The method was tested on a turbofan engine degradation simulation dataset provided by NASA.

## Decision Trees, Neural Networks, & Model Optimization

Previous research has discussed the mapping of DT into a multilayer NN structure as the design for a class of layered NN known as Entropy Nets [32]. Research detailed a number of important issues including automatic tree generation, the integration of incremental learning, and the generalization of knowledge obtained during the tree design phase. The research presented the number of neurons required in each layer of the NN and the desired output, therefore promoting a faster progressive training procedure which enables each layer to be trained independently.

Another research effort compared the efficacy of particle identification in physics through Artificial Neural Networks (ANN) and Boosted Decision Trees [33]. On the basis of studies of Monte Carlo samples of simulated data, the research found that boosting algorithms performed better than ANN for particle identification. In other works, prediction of electricity energy consumption and sound pressure level was analyzed using traditional regression analysis, DT, and NN [34, 35].

A study was performed on the automatic analysis and classification of attribute data from training course web pages by comparing the performance of NB, DT, and NN [36]. The work presented an ensemble NB classifier supplemented by a “believed probability” algorithm which was compared to DT and NN classifiers when classifying the training course domain. Results indicate that the ensemble NB classifier outperformed the DT and NN classifiers in most metrics due to the supplemental algorithm.

A recent study integrated Principal Component Analysis (PCA) with deep NN to predict multiple decay state coefficients in naval propulsion systems [37]. Prediction results were compared with different numbers of hidden layers to analyze the impact of hidden layer

architecture on the performance. Another study predicted aerofoil self-noise at an early stage of design using NN and a hybridization of PCA with NN [38]. Results were compared between NN, PCA – NN, and different regression techniques, and demonstrated that the PCA – NN outperformed all other techniques. Another recent work proposed a method of communication between specialized NN [39]. The method consisted of numerous specialized sets of NN each trained for specific tasks which would communicate by transferring knowledge between themselves. This allows the NN to evolve by chaining the input and output information. The method allows different NN to be plugged in to the loop for knowledge transfer to evolve. Additional information can be requested if the current task is difficult to resolve based on the already present information. This method is known as Outward Inward Neural Network and Inward Outward Neural Network Evolution (OINNIONN) and, as the method can transfer the learning model, is applicable to aircraft maintenance.

Most prior research compared classification methods such as NN, DT, and linear discriminant analysis [40 – 43]. Analysis of variance is used to identify any significant differences between the results of the methods. The issues of finding the most appropriate network size and using an independent validation dataset to determine when training the network should cease are also discussed. However, the integration of DT and NN as a unique classification method for aircraft maintenance has not been described in the literature.

Taking the vast literature of aircraft maintenance, ML, ensemble learning, and hybrid modeling into consideration, this work proposes a unique classification technique whereby each node of a DT is integrated with a specialized NN and applied to aircraft maintenance data.

## **METHODOLOGY**

### Uniqueness of Problem

Compared to many other classification problems, the issue of determining a cause of an aircraft accident or incident usually depends on multiple attributes. Furthermore, any solution found consisting of a classification method would not identify the cause but identify the main feature, or subsystems, which contributed to the accident or incident.

The problem of classification assumes that many values of the data are missing. In addition, certain records have fields which might have been mislabeled due to erroneous handling or just to save time during the recording of an incident.

Last but not least, an error which leads to an accident could easily cause a large number of casualties. Since the aircraft types are shared by multiple organizations, the chance of a rare incident reoccurring is relatively high. Therefore, the classification of an incident at an early stage is critical.

### Problem Setting

Since we aim to use a DT and multiple NN for learning to classify incidents and accidents, the first step is quantifying the input in numeric values. Many of the inputs are formatted as textual labels or free text affiliated with the value such as: Aircraft Make, Aircraft Model, and Part Name. These labels for each input attribute were assigned a numerical value to represent all possible labels. The numerical value was designed to have a uniform distribution  $U(0, 1)$ , with -1 representing no value.

The next issue was to define the depth of each NN in each node of the DT. Due to the success of deep NN in multiple domains, we analyzed how deep the NN should be to optimize the results. We analyzed how many hidden layers,  $n$ , are required to optimize the neural network performance. Although theoretically it could be assumed the deeper the better, in reality, we show in the experiments that the optimum number of hidden layers is reached fairly quickly at 3-4 layers. Furthermore, after a fixed number of added hidden NN layers, the results suddenly drop to be equivalent to guessing. In our experiments, adding any additional hidden layers above 29 results in a drop to 50% of the performance results measured, which is equivalent to guessing in a binary DT.

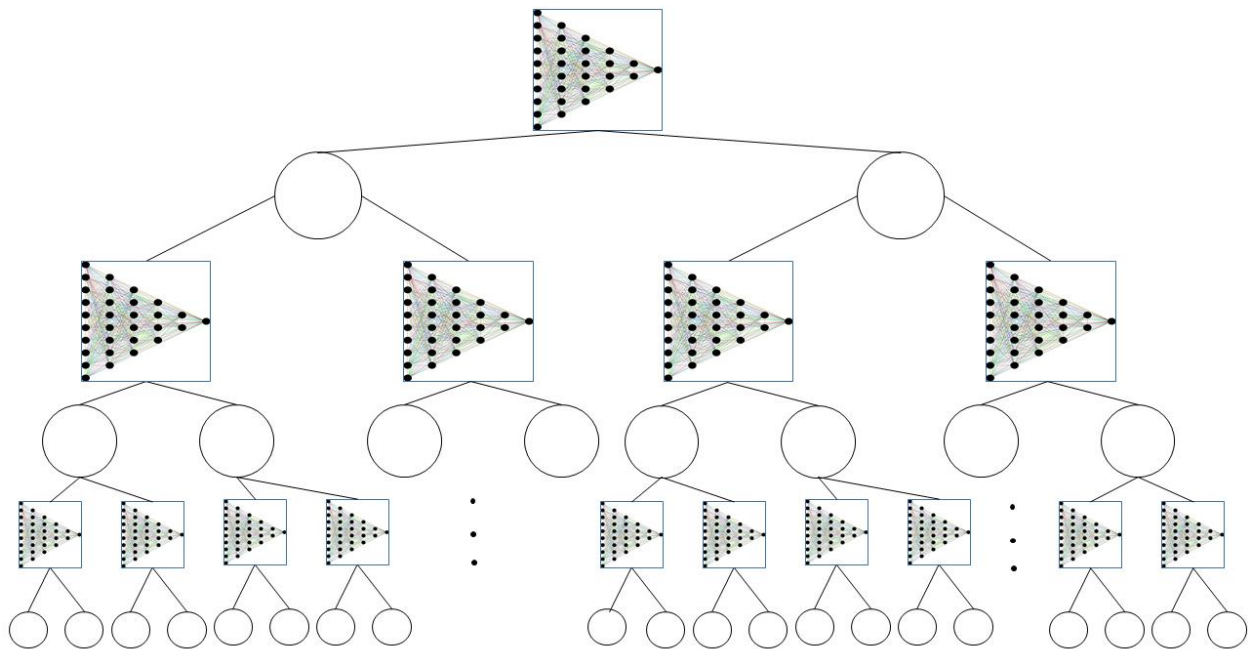
Theoretically, the number of inputs in the neural network should be equal to the number of variables which are available. The assumption is that the NN can learn to ignore the attributes which do not contribute to the optimization of the solution. In reality, these are two different tasks which should be handled by different NN:

- Classifying the important contributing input variables.
- Optimizing a single decision in a classification.

For classifying the important contributing input variables, a NN was trained using all 72 inputs. Once the NN results have converged to a fixed value, we evaluate the weights. The weights between the input layer and the first hidden layer represent the importance of each input variable. These weights were recorded for each input variable. Then the absolute value of every weight is taken, and the average of these absolute values is recorded for each variable. The inputs with the highest averages are determined to be the significant inputs. Input variables with low mean weight

value have less contributing effect to the optimization of the classification and therefore were removed.

The experiments show that limiting the number of input variables contributed to the performance of the classification. The best number of input variables can be determined by organizing the input variables in descending absolute average weight order and either adding or removing one variable at a time until there is a change in the output performance. It should be noted that a large mean weight difference does not always correlate to a large difference in the performance. However, the descending order of the mean weight is an important factor contributing to the output performance.



*Fig. 1: Integration of Decision Tree & Neural Network Where Each Classification Node is a Specialized Neural Network*

## Integrating Decision Trees and Neural Networks

Next, we aim to optimize a single decision in a classification. We integrate the decision tree approach with neural networks. We create a decision tree with a neural network at each node of the decision tree, displayed in Fig. 1.

The unification of the DT and NN approach allows us to integrate the advantages of both methods. The NN works well while classifying into categories where the boundaries of classification are less distinct, but performance drops when there is a large number of categories. The DT works with a large number of categories which are distinctly classified.

The DT is built based on a set of possible results which can occur (accidents or incidents). For this, we choose the best possible result attribute with the highest information gain. To define information gain, we define a measure commonly used in information theory, called entropy, which characterizes the (im)purity of an arbitrary collection of examples [44].

**Entropy**  $H(S)$  is a measure of the amount of uncertainty in the dataset.

$$H(S) = -\sum_{c \in C} p(c) \log_2 p(c)$$

Where,

$S$  – The dataset for which entropy is being calculated in the current iteration.

$C$  – The set of the classes in  $S$ ,  $C = 0, 1$ .

$p(c)$  – The proportion of the total elements in class  $c$  to the total elements in set  $S$ .

If  $H(S) = 0$  then the set  $S$  is perfectly classified

**Information Gain**  $IG(A)$  is the measure of the difference in entropy from before to after the set  $S$  is split on a result attribute  $A$ . This measures how much of the uncertainty  $S$  was reduced after splitting set  $S$  on result attribute  $A$ .

©

Where,

$H(S)$  – Entropy of set  $S$ .

$T$  – The subsets created from splitting set  $S$  by result attribute  $A$  such that

©

$p(t)$  – The proportion of the number of elements in  $t$  to the number of elements in  $S$ .

$H(t)$  – Entropy of subset  $t$ .

The information gain can be calculated for each remaining attribute. The attribute with the largest information gain can be used to split the set  $S$  on each iteration.

After selecting the attribute with the largest information gain, we build a NN based on the criteria discussed in the previous section (*Problem Setting*). For each maintenance problem, we construct a NN which is designed to classify only if the problem occurs. Each NN at each node of the DT consists of all the result attributes which could lead to a possible accident or incident. It should be noted that the result attributes represent the problem and are different from the input attributes filtered in the previous section.

Each leaf of the DT includes a NN with a binary classification task which improves its performance. Since each NN is tailored to a specific classification, the overall performance of the system does not depend on the performance of a single neural network.



The actual implementation does not necessarily require the implementation of NN for all possible problem attributes since many categories of problems in the area of maintenance can be classified under one classification category. Furthermore, a maintenance investigator can sometimes easily identify the correct cause at a higher level of the classifications.

## Model Outline

The structure of the hybrid binary tree classifier gives us the flexibility to find more detailed or less detailed classification problems. The structure is formatted in a way so that the height of the tree represents the complexity of the prediction of the problem. The larger the tree, the more detailed an issue that the network can predict on. Each level of the tree is trained on different data that is pulled from the original dataset. If the first node in the tree was predicting for a crack in the airplane, it would assign the original dataset with a crack problem a 1 and all of the other entries without a crack problem 0. If we continued the tree to predict for something with a crack and a fuselage problem, we would separate the original dataset into two datasets, one with crack data and one without crack data, and assign each entry with a fuselage problem a 1 or a 0 and train a new model for each of the two subcategories. Since we split the dataset multiple times, this method would work best in an environment where there is a lot of data. This method of predicting for crack then predicting for fuselage in the binary tree format actually performs better than just predicting for entries that contain crack and fuselage or not from the beginning.

## Data Preprocessing

Before the data is fed into the NN, it is first preprocessed so that it may be interpreted by them. The preprocessing work was performed within the Python programming language along with the Pandas data analysis package through a multi-step process. First the data is read from a

Comma Separated Value (CSV ) file in chunks, approximately 5000 records at a time, to a Pandas data frame. The purpose of chunking the data rather than processing it all at once is to conserve memory resources, allowing the preprocessing method to be scaled to data of great volume. Next, a set of dictionaries is created, with one dictionary corresponding to one column/variable in the data, for mapping unique, non-numeric values in each variable to a numeric identifier. Then each column is parsed for unique values. If a value is encountered which is not in the dictionary for a given variable, then it is added to the dictionary along with a numeric identifier, and the identifier is incremented. This starts at an identifier value of 100 and increases by 100 for each unique value found in a variable so as to adequately space the numeric values apart when normalizing the data later. However, when a null value is encountered, it is assigned a value of -1 instead to separate it from non-null data. Once each variable has been parsed in a chunk, the dictionaries are used with a mapping function to convert all the non-numeric values in that chunk to their numeric identifiers. After this, the chunk is then written to a new CSV file with a similar name to the unprocessed CSV file to preserve the original unprocessed data. If more data exists in the CSV file, then another chunk is read and the above process repeats. The dictionaries for each variable are preserved across chunks to maintain consistency in mapping the data. The final step in the preprocessing stage, once all of the chunks have been converted, is the creation of a text file containing variable name headers and all of the numeric identifiers for each variable and which value each identifier represents. This allows both the user and the software to determine which identifier maps to which unique value for any given variable.

Some variables are skipped entirely in the preprocessing stage if they are numeric, and it has been decided that their numeric values are significant. We chose these variables to be skipped as their numeric values would lose meaning if mapped to an arbitrary integer value. For example:

the variable *Aircraft Total Time* represents the total amount of time the aircraft has been in use. We determine this to be a potentially significant variable for predicting aircraft incidents. However, if we were to map a value of 3,892 hours to an integer of 300 and a value of 1,765 hours to an integer of 700 then the magnitude of usage time could lose its value. Additionally, as it is unlikely that no two aircraft will share the same amount of flight time mapping these values to unique integers could also add unnecessary complexity to the data with the volume of unique integer values.

## **EXPERIMENTS & RESULTS**

### Data

The Federal Aviation Administration collects all preliminary accident and incident information reported to the Office of Accident Investigation and Prevention. The data includes accident and incident data categorized by the aircraft manufacturer. The experiments focused on the Boeing 737 dataset.

The dataset contains 137,236 records with each record consisting of 73 variables. These records included data from aircraft that suffered mechanical issues. We chose this data for two main reasons. The first reason is because it is a real-world dataset that was hand documented for actual maintenance operations. Demonstrating that this algorithm can be successfully applied to a hand recorded dataset shows the robustness of the given algorithm. The second reason is because of the sheer size of the data that we train our algorithms on. Having a dataset which contains a large number of real-world records allows for us to make sure that the algorithm is able to handle complex inputs and perform with high accuracy.

Of the 73 total variables in the dataset, 72 variables were used as inputs to each of the NN. Each NN had a single neuron classifying whether the record belongs to a specified category. The data was split into 75% training, 15% testing, and 10% validation. The testing data is used to test the accuracy and F1 score, basically a weighted average between precision and recall of the NN. The F1 score measure is chosen as a better representation of our model's performance as it considers the precision and recall values rather than correct predictions as Accuracy does. The validation data ensures that there is no overfitting.

Table 1: Boeing 737 Dataset Composition

Operator Control Number	Difficulty Date
Submission Date	Operator Designator
Submitter Designator	Submitter Type Code
Receiving Region Code	Receiving District Office
SDR Type	JASC Code
Nature Of Condition A	Nature Of Condition B
Nature Of Condition C	Precautionary Procedure A
Precautionary Procedure B	Precautionary Procedure C
Precautionary Procedure D	Stage Of Operation Code
How Discovered Code	Registry N Number
Aircraft Make	Aircraft Model
Aircraft Serial Number	Aircraft Total Time
Aircraft Total Cycles	Engine Make
Engine Model	Engine Serial Number
Engine Total Time	Engine Total Cycles
Propeller Total Time	Propeller Total Cycles
Part Make	Part Name
Part Number	Part Serial Number
Part Condition	Part Location
Part Total Time	Part Total Cycles
Part Time Since	Part Since Code
Component Make	Component Model
Component Name	Component Part Number
Component Serial Number	Component Location
Component Total Time	Component Total Cycles
Component Time Since	Component Since Code
Fuselage Station From	Fuselage Station To
Stringer From	Stringer From Side
Stringer To	Stringer To Side
Wing Station From	Wing Station From Side
Wing Station To	Wing Station To Side
Butt Line From	Butt Line From Side
Butt Line To	Butt Line To Side
Water Line From	Water Line To
Crack Length	Number Of Cracks
Corrosion Level	Structural Other
Discrepancy	

Table 1 details the variable composition of the dataset. In addition to the 72 variables used as input, the last variable, *Discrepancy*, contains a free text description of the accident or incident. The *Discrepancy* field was used as the output variable to be classified by parsing the free text description for categorical keywords to classify each accident.

## Methods

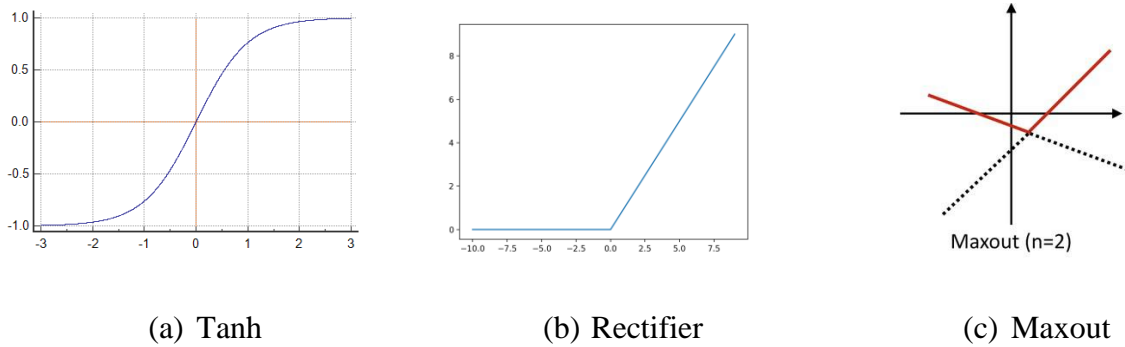


Fig. 2: Activation Function Used

The following activation functions were used in the experiments:

*Tanh* – Hyperbolic Tangent Function (Fig. 2a).

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

*Tanh w/ Dropout* – Tanh with a dropout ratio of 0.5 for each hidden layer.

*Rectifier (default)* – Positive part of its argument (Fig. 2b).

$$\max(0, x)$$

*Rectifier w/ Dropout* – Rectifier with a dropout ratio of 0.5 for each hidden layer.

*Maxout* – Given an input  $x$ , a maxout hidden layer implements the function (Fig. 2c).

$$\max(w_1 x + b_1, w_2 x + b_2)$$

where  $w_1$ ,  $w_2$ ,  $b_1$ , and  $b_2$  are learned parameters.

*Maxout w/ Dropout* – Maxout with a dropout ratio of 0.5 for each hidden layer.

Multiple NN configurations were analyzed for best performance. For the following experiments, a NN with 3 hidden layers each containing 60, 40, and 20 neurons respectively was used.

## Neural Network Pseudocode

The NN models were constructed, trained, and evaluated using Python and the H2O.ai ML framework. The following Table 2 contains pseudocode representing this process.

*Table 2: Neural Network Pseudocode*

---

- 1: *CSVData*: Data read in from a preprocessed CSV file.
- 2: *trainData*: Subset of *CSVData* used for training NN.
- 3: *testData*: Subset of *CSVData* used for testing NN.
- 4: *validData*: Subset of *CSVData* used for validating NN.
- 5: *nn*: H2ODeepLearningEstimator NN model.
- 6: *nnMetrics*: H2O data frame containing performance metrics from testing the NN.
- 7: *hiddenLayers*: Matrix containing the hidden layer structure of the NN.
- 8: *activationFunction*: String value of the activation function to use.
- 9: *invars*: List of input variables from the data.
- 10: *outvars*: List of output variables from the data.
- 11: *resultsCSV*: CSV file for containing NN performance metrics.
- 12: Import H2O, H2ODeepLearningEstimator
- 13: *CSVData* = *open*("csvfile.csv"; "read")
- 14: H2O.*init*()
- 15: H2O.*read*(*CSVData*)
- 16: *trainData* = 75% of *CSVData*
- 17: *testData* = 15% of *CSVData*
- 18: *validData* = 10% of *CSVData*
- 19: *nn* = *DeepLearningEstimator*(*hiddenLayers*, *activationFunction*)
- 20: *nn.train*(*invars*, *outvars*, *trainData*, *validData*)
- 21: *nnMetrics* = *nn.test*(*testData*).*performanceMetrics*
- 22: *resultsCSV* = *open*("results.csv"; "write")
- 23: *resultsCSV.write*(*nnMetrics*)
- 24: *resultsCSV.close*()

---

## Experiments

The dataset is first preprocessed to conduct the experiments. The Pandas library in Python is used for preprocessing and simulation experiments. The strings in each column/variable of the dataset are parsed and mapped to an integer value. The starting value is selected as 100 and is increased by 100 for every unique subsequent string. This process is repeated for every variable individually. However, in the case of an integer input or floating-point variables, magnitude is important (such as the total flight time of a 737), and the values are not mapped for that variable and are simply skipped. For all variables, a null value is mapped to -1. The data set is labeled in the present study. While classifying the inputs, we used the significant inputs from the root of the DT where only “maintenance” or “non-maintenance” was classified. Through further experimentation we determine whether the significant input is changed at each node of the tree.

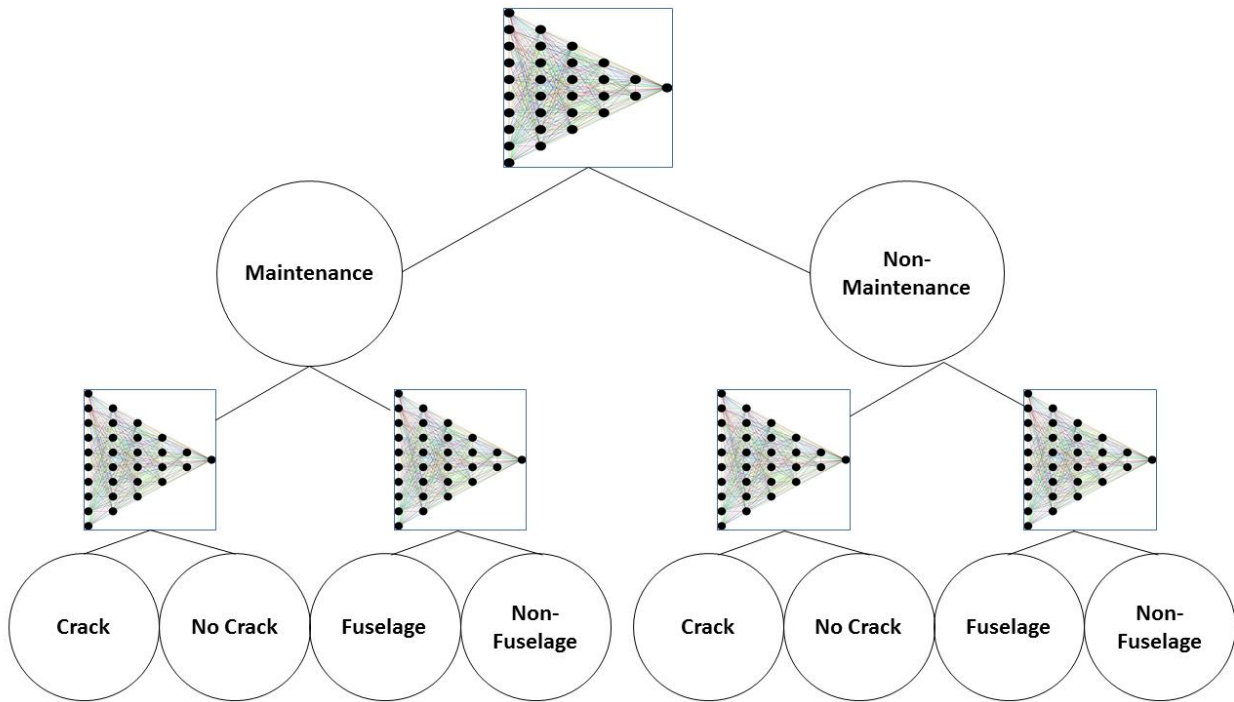
The dataset is classified into two categories: whether the problem with the aircraft occurred during maintenance or not during maintenance. Thereafter, the data is further classified into whether or not the problem involved cracks and whether or not the problem involved the fuselage. In this way, by classifying fuselage after classifying maintenance, we mean that we first identified that the problem occurred during maintenance and then identified that the problem involved the fuselage. These two subcategories regarding the fuselage or presence of cracks were chosen as they appeared to have high support within the free text descriptions of *Discrepancy*.

The first set of experiments analyzed how deep the deep NN should be. We analyzed a binary classification of accident or incident identification during Maintenance or Non-Maintenance. We increased the NN hidden layers from 1 to 100 and checked how the F1 and accuracy results change. The Stopping tolerance = 0.0000001 was used for all of our experiments.



This precise value of stopping tolerance ensures convergence of the NN with higher performance. These experiments analyzed what the correct structure of the NN would be.

The second set of experiments analyzed whether a larger dataset, would correlate with better results. We organized the input variables in descending order of the mean value of the weight connecting the input layer and the first hidden layer. We then increased the number of input variables used in descending order of weight value and compared the Area Under the Curve (AUC), Accuracy, Precision, Recall, and F1 values. Each of these values was compared with the six types of activation functions.



*Fig. 3: Decision Tree - Neural Network with Specific Classification Categories*

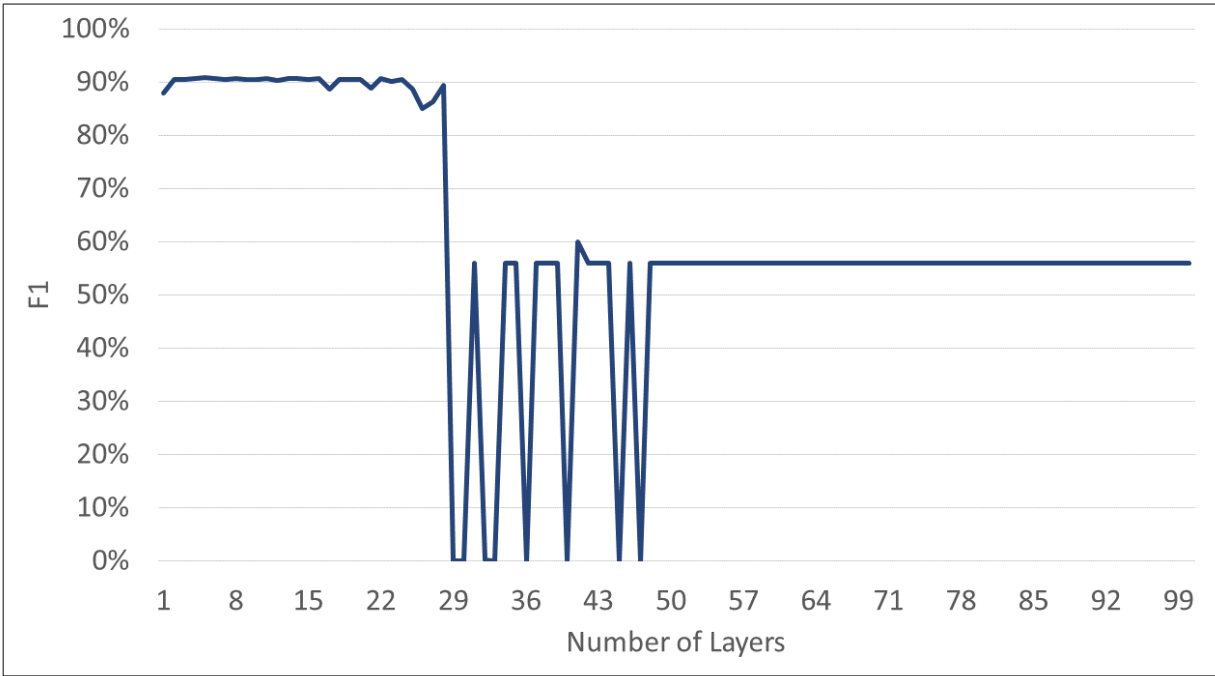


Fig. 4: F1 Score vs. Number of Hidden Layers

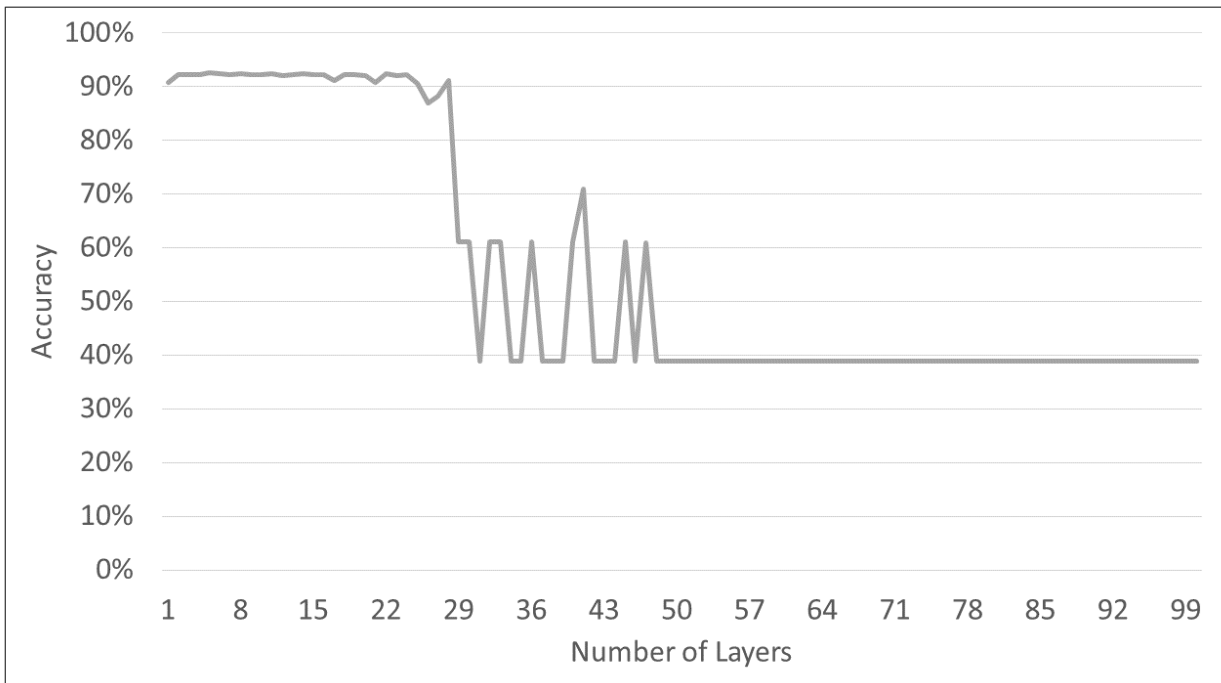


Fig. 5: Accuracy vs. Number of Hidden Layers

The third set of experiments analyzed the advantages of integrating the DT and the NN methods. An outline of the set of experiments performed is described in Fig. 3. First a neural network was used to perform a binary classification into categories Maintenance or Non-Maintenance. Each of the classified records was then again classified into Crack or No Crack and Fuselage or No Fuselage. The classification into each of these subcategories was performed using all the data previously classified into the main category. In other words, a record from Maintenance and Crack could also belong to either Fuselage or Non-Fuselage but not to both. As can be seen from Fig. 3, four different NN were used to classify to the eight different sub-classifications.

## Results

Fig. 4 and Fig. 5 display results of the analysis of the appropriate depth of the deep NN. Results peak at three hidden layers and continue around the same F1 (Fig. 4) and Accuracy (Fig. 5) levels until 29 hidden layers. From this point onward, the results show that the network would be too deep. The network results show that over 29 hidden layers is equivalent to guessing in a binary classification. The F1 value becomes slightly above 50% and the Accuracy slightly below 50%. The sharp peaks in Fig. 4 and Fig. 5 represent numbers of hidden layers where the NN becomes unstable in classifying the data, resulting in errors when calculating accuracy and F1 score. The drops to 0% in Fig. 4 exactly represent these errors as a value of 0 was returned when an error in calculating F1 score occurred. It seems that a three hidden layer NN is accurate and fast enough to perform the task of classification. For the experiments that we performed the maximum amount of time that the networks took to train the neural models spanned from 20 to 30 seconds for each network.

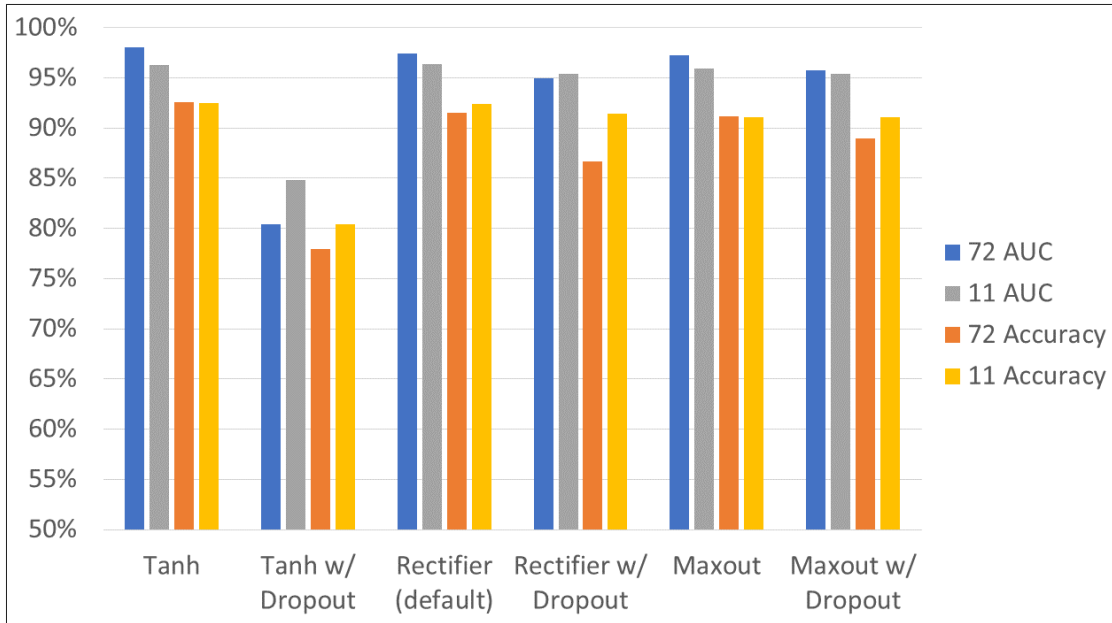


Fig. 6: Comparing the Performance of Activation Functions. Here, the Values of AUC and Accuracy are Measured When Using All 72 Inputs and When Using Only the Identified 11 Significant Inputs for Each Activation Function.

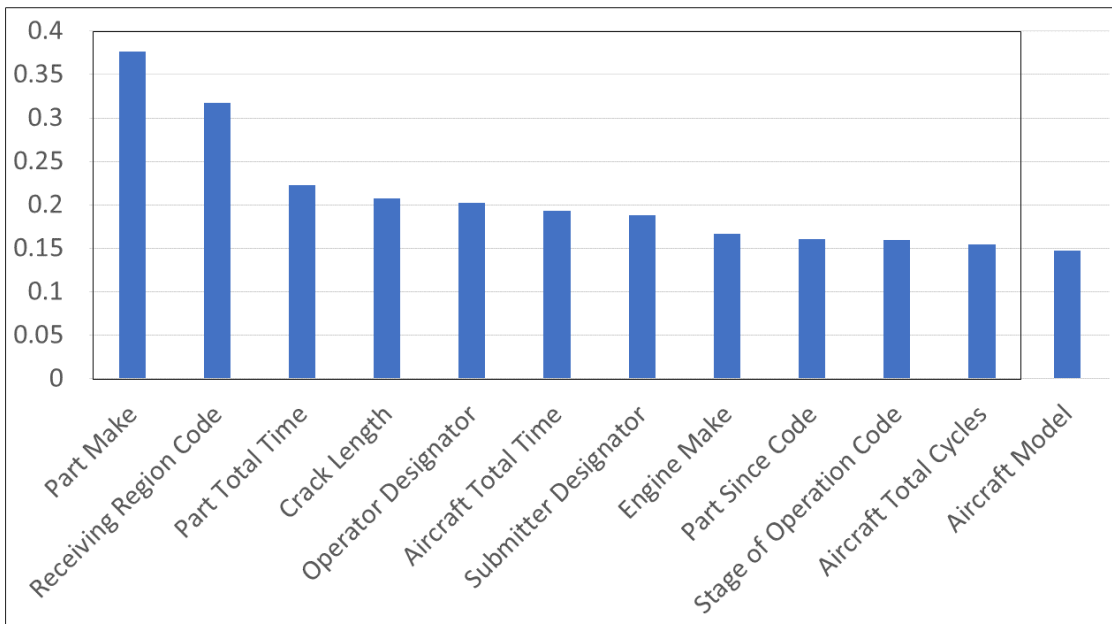
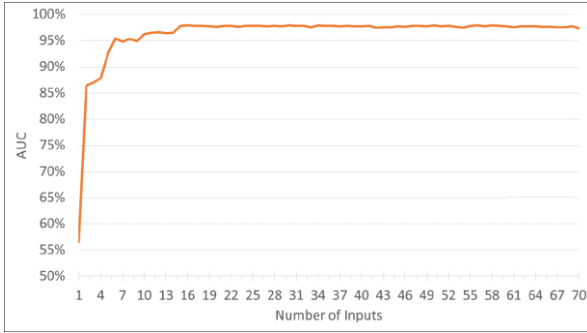
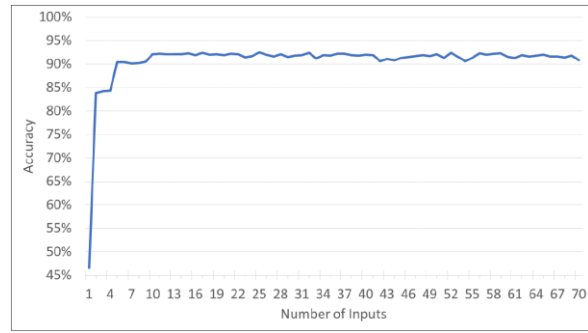


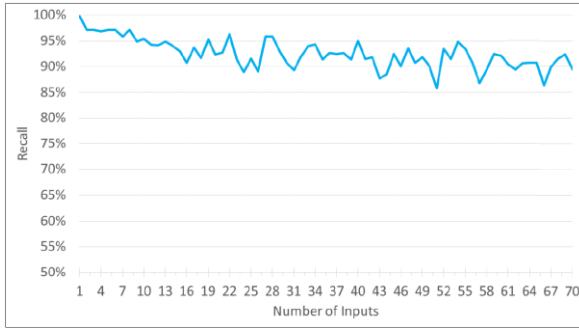
Fig. 7: Average Weights for Leading First Layer Inputs. These Bars Represent the Highest Absolute Value Average Weights Between the Input Layer and First Hidden Layer. The First 11 Bars Represent the Significant Inputs.



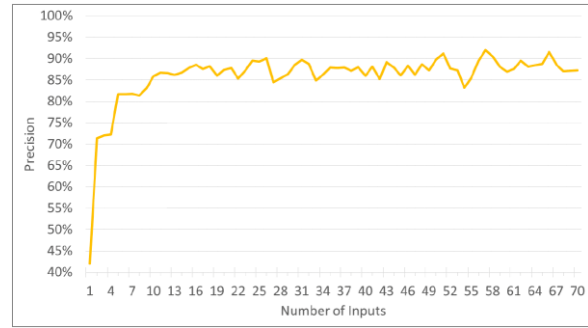
(a) AUC vs. Inputs



(b) Accuracy vs. Inputs



(c) Recall vs. Inputs



(d) Precision vs. Inputs

Fig. 8: AUC, Accuracy, Recall, and Precision vs. Number of Inputs.

Fig. 6 presents the classification results into the Maintenance and Non-Maintenance categories as the number of input variables increases. Fig. 6 shows the AUC and Accuracy of all six activation functions comparing the results of using only the top 11 mean weight variables versus using all possible 72 variables. The results show the accuracy is almost the same, -0.12%, and up to 4.77% better when using only the top 11 variables. The AUC is less consistent and varies from -1.76% to 4.44% for using the top 11 identified variables versus all 72. The results show the advantage of the method of identifying the top variables before using the NN as a classifying tool.

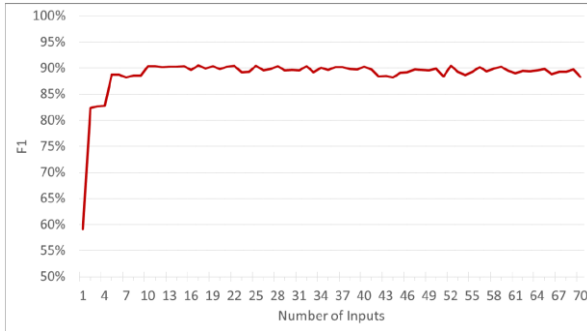
Fig. 7 shows the average mean for the leading inputs weight value between the input layer and the first layer. These identify the main variables which are relevant for high accuracy results.

The top 11 variables appear in the circumference box. The results show that issues such as Part Make, Receiving Region Code, and Part Total Time can clearly be identified as the most relevant classifiers. The list of the leading main contributors for the accident and incident reports ends with Aircraft Total Cycles. The Aircraft Model is already identified as a less unique classifier for the type of issue involved. These values included the different models of the Boeing 737.

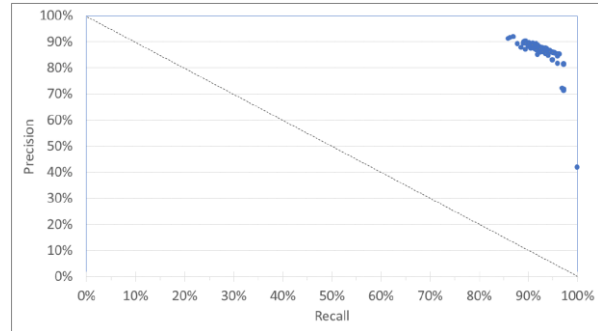
Fig. 8a presents the AUC as the number of inputs increases. Fig. 8b presents the Accuracy as the number of inputs increases. The inputs in the x-axis are arranged in descending order of mean weights leading from the input layer of the NN to the first hidden layer. In other words, the most significant input is added first, and each subsequent input variable added possesses a lower average weight between the input layer and the first hidden layer, i.e., each subsequent input is less significant than the previous. The AUC continues to increase as the number of inputs increases up to 16 inputs. However, the accuracy does not improve over 11 inputs which were identified as the important variables.

The AUC difference can be viewed as a less accurate value for measuring performance. In this case, it can be attributed to the low number of values measured to create the curve. This could explain the difference when measuring the area with AUC versus comparing a single Accuracy result.

Similarly, Fig. 8c presents the Recall and Fig. 8d presents the Precision as the number of variables with the descending mean weight increases. These results display that the recall actually declines, from 100% to above 85% as more variables are added. However, the precision increases and stabilizes after the top 11 weighted variables are included. The results show that the recall has a slight drop as more variables are added. However, the precision is determined by the leading or “more important” variables.



(a) F1 Score vs. Inputs

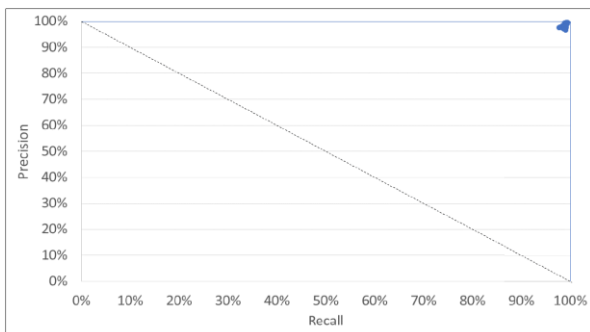


(b) Precision vs. Recall

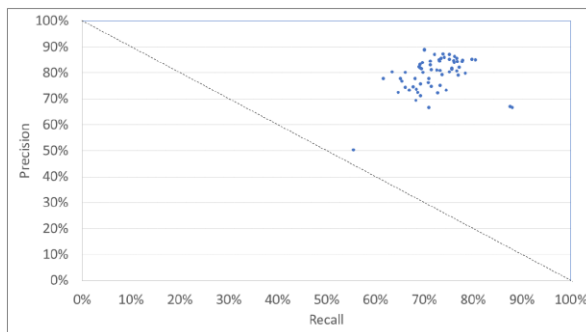
*Fig. 9: F1 Score vs. Number of Inputs, Precision vs. Recall.*

These results can be viewed more clearly when viewing the F1 value appearing in Fig. 9a. As the number of highly weighted variables is added the value peaks up to 11 variables. From 11 variables the F1 is stable at around 90%. Furthermore, the Precision vs. Recall in Fig. 9b shows that most results are clustered in the top right except for the initial values with high recall.

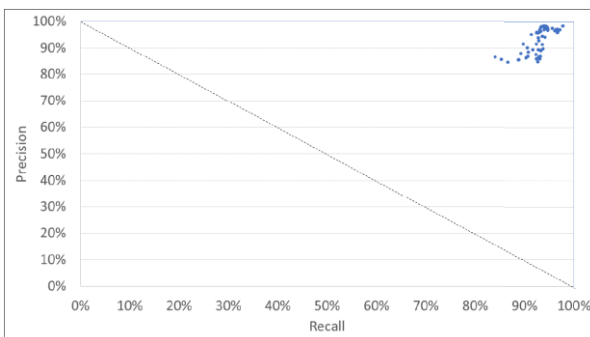
The results show the correct identification of the important inputs by the method of classifying mean weights in descending order. The additional input variables which do not seem to improve the results can be attributed to constant values, variables which are dependent on other inputs, or values which are inconsistent with the expected results.



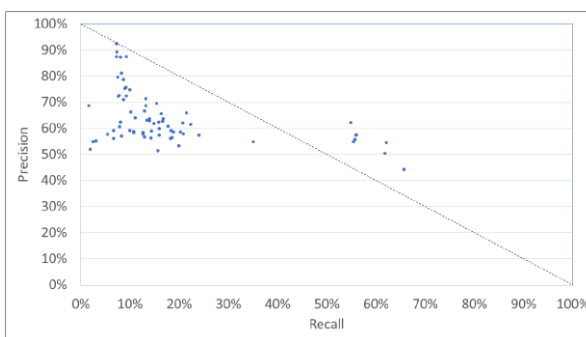
(a) Precision vs. Recall for Maintenance and Crack



(b) Precision vs. Recall for Maintenance and Fuselage



(c) Precision vs. Recall for Non-Maintenance and Crack



(d) Precision vs. Recall for Non-Maintenance and Fuselage

Fig. 10: Precision vs. Recall for Maintenance, Crack, and Fuselage Classifications.

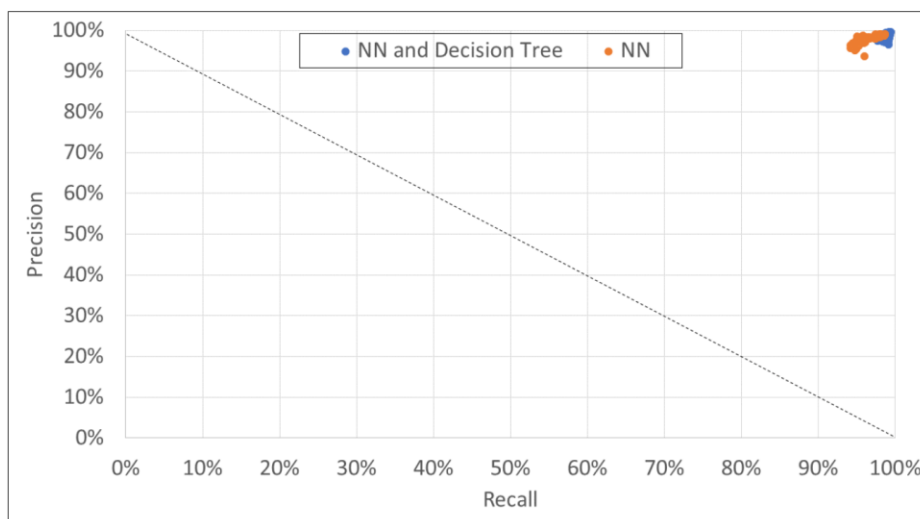


Fig. 11: Precision vs. Recall Comparison Between DT - NN Model and NN.



On the other hand, Fig. 10b and Fig. 10d show what happens when the DT and NN are not aligned correctly. In this case, classifying Fuselage after classifying Maintenance has slightly lower precision versus recall results. However, the classification of Fuselage after the classification as Non-Maintenance has been performed is centered around the diagonal, which represents guessing in a binary classification. This shows the incorrect DT structure. One less likely possible explanation is that all Fuselage classifications are only identified during Maintenance. Another, more likely, option is that this part of the DT is not properly constructed. In other words, Fuselage under Non-Maintenance cannot be classified. This means that the dataset did not have a sufficient number of cases where problems with the fuselage occurred during a time when the aircraft was not undergoing maintenance, and therefore the present method could not accurately be classified with our current methods.

At least one more layer of sub-classification needs to be added in order to correctly identify this issue. Another concept should be added to the DT below Non-Maintenance before trying to identify whether there is a Fuselage problem.

Finally, Fig. 11 shows the advantage of our hybrid method integrating the NN with the DT compared to the commonly used method which uses just NN for classification. The figure shows the precision and recall as the number of inputs increase. The DT – NN method outperforms the method of using only NN for both precision and recall.

## Activation Function & Hidden Layer Tests

Tests were performed to determine how different activation functions and hidden layer architecture contributed to the overall performance of the NN, which was measured through Accuracy, AUC, Precision, Recall, Precision – Recall AUC, and F1 scores. The activation

functions used were Rectifier, Tanh, Maxout, Rectifier w/ Dropout, Tanh w/ Dropout, and Maxout w/ Dropout, where each of the dropout rates was 0.5. The hidden layer architectures all consisted of three layers with differing quantities of neurons in each layer. The layers began with [40, 30, 10] (the default architecture used for most other experiments due to its high performance) and subsequent architectures tested followed the pattern: [40, 30, 9], [40, 30, 8], . . . , [40, 30, 5], [40, 25, 10], . . . , [40, 5, 5], [35, 30, 10], . . . , and ended with [10, 15, 5]. These architecture tests were conducted alongside the activation function tests so that each architecture was tested with each of the six activation functions. The tests concluded that a NN using the Tanh activation function performed better than the other activation functions used, with each of the dropout functions performing the worst. The best test resulted in a 92.5% accuracy and a hidden layer of [35, 15, 9].

## Pearson/Spearman Correlation Tests

Pearson and Spearman correlation tests were also performed to find statistical correlation between the variables when comparing them to the Discrepancy column as well as every other input variable. Pearson correlation was used to identify any linear change relationships between the input variables while Spearman correlation was used to identify monotonic relationships between the input variables. These tests were run on the enumerated data. Aircraft Total Cycles was shown to be the most significant variable with both the Pearson and the Spearman test. These results could be implemented/applied in the future to find out which significant inputs could be used for training the NN.

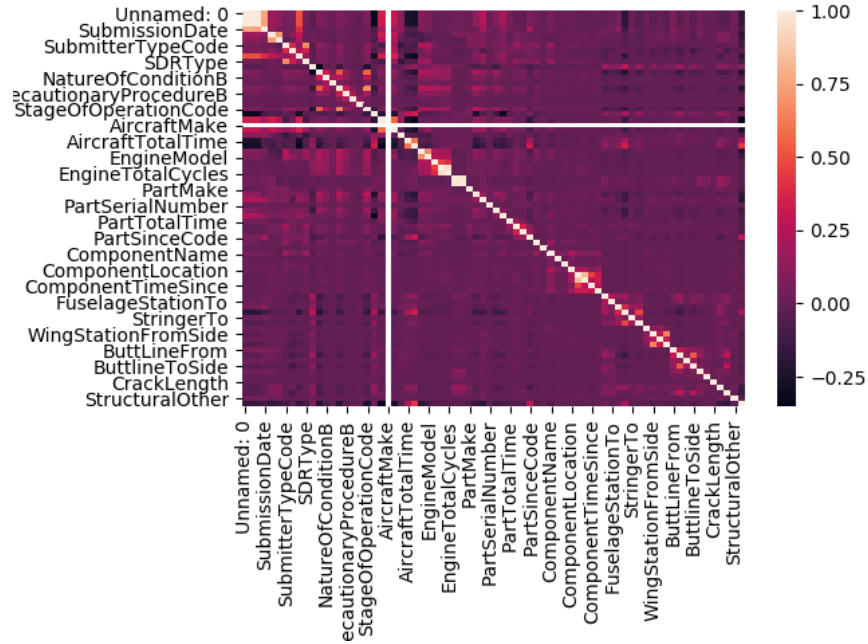


Fig. 12: Pearson Correlation Heatmap for Crack

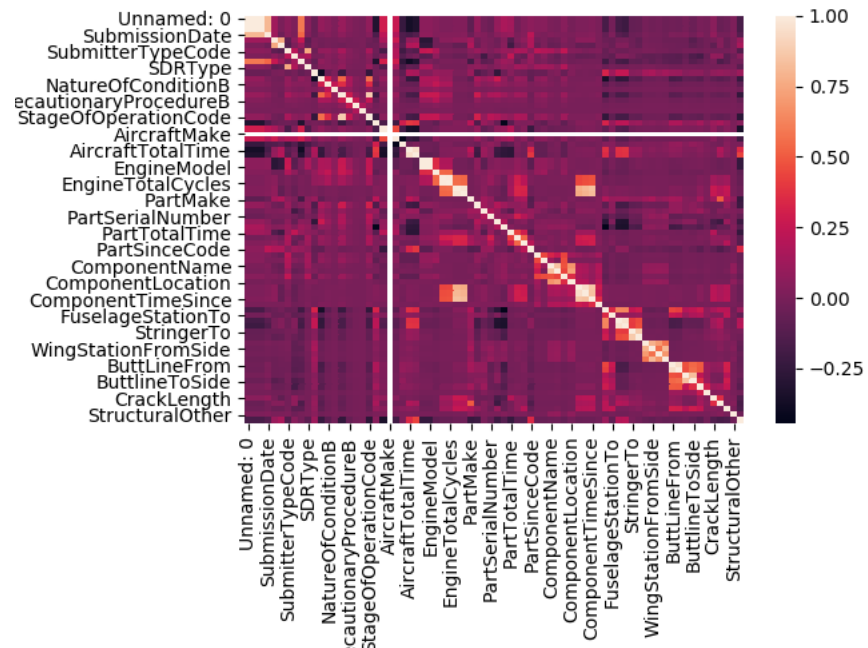


Fig. 13: Spearman Correlation Heatmap for Crack

Table 3: Pearson Correlation Between Discrepancy and Inputs

Variable Names	Discrepancy
OperatorControlNumber	-0.162920734
OperatorDesignator	-0.191397581
SubmitterDesignator	-0.139558448
SubmitterTypeCode	-0.114425491
ReceivingRegionCode	-0.030500605
JASCCode	0.098274881
StageOfOperationCode	-0.238238549
HowDiscoveredCode	0.032754564
RegistryNNumber	-0.351326718
AircraftModel	-0.248164469
AircraftSerialNumber	-0.061595756
AircraftTotalTime	0.225559051
AircraftTotalCycles	0.425399651
EngineMake	-0.217505143
PropellerTotalCycles	-0.035912637
PartSerialNumber	-0.015229973
PartTotalTime	-0.084394422
PartTimeSince	-0.062439797
PartSinceCode	0.28765802
ComponentSinceCode	-0.017284174
FuselageStationFrom	0.029334189
FuselageStationTo	-0.028950023
StringerFrom	0.009530162
StringerFromSide	0.19310386
CorrosionLevel	-0.148433297

Table 4: Spearman Correlation Between Discrepancy and Inputs

Variable Names	Discrepancy
OperatorControlNumber	-0.162920813
OperatorDesignator	0.106873514
SubmitterDesignator	0.158101481
SubmitterTypeCode	-0.129345355
ReceivingRegionCode	0.048436281
JASCCode	0.118254811
StageOfOperationCode	-0.271904763
HowDiscoveredCode	0.048619706
RegistryNNumber	-0.340778447
AircraftModel	-0.199877064
AircraftSerialNumber	0.010545024
AircraftTotalTime	0.34909541
AircraftTotalCycles	0.43531384
EngineMake	-0.236875842
PropellerTotalCycles	-0.035912637
PartSerialNumber	-0.071002955
PartTotalTime	-0.239128945
PartTimeSince	-0.117941956
PartSinceCode	0.341907897
ComponentSinceCode	-0.021837315
FuselageStationFrom	0.111152493
FuselageStationTo	-0.0815809
StringerFrom	0.188622223
StringerFromSide	0.206177495
CorrosionLevel	-0.151251426

## Heatmap and Correlation Information

The heatmaps that are represented in Fig. 12 and Fig. 13 demonstrate the correlation of every variable to each other using the Pearson/Spearman statistical correlation algorithm to find out how closely associated each variable is with another. The closer the number is to 1, the higher the positive linear correlation is with the variable being compared and the lighter the area is in the heatmap. The closer the number is to -1, the higher the negative linear correlation is with the variable being compared and the darker the area is on the heatmap. When the number is zero, it means that there is no linear correlation between the two variables. So, the closer the number is to the more association the variables have with each other.

This extends to Table 3 and Table 4 which illustrate the correlation between a sample of 25 inputs and Discrepancy. The Pearson correlation results between all input variables and

Discrepancy in Fig. 14 show a linear correlation coefficient of 0.15 or greater between the input variables AircraftTotalCycles, PartSinceCode, AircraftTotalTime, and StringerFromSide. The Spearman test shown in Fig. 15 shows a correlation coefficient of 0.15 or greater for the input variables AircraftTotalCycles, AircraftTotalTime, PartSinceCode, StringerFromSide, StringerFrom, and SubmitterDesignator. Analyzing the results of the combined tests reveals that AircraftTotalCycles, PartSinceCode, and AircraftTotalTime are the three variables with the highest degree of correlation. AircraftTotalCycles was shown to be the most significant variable with both Pearson and the Spearman test. These results could be implemented/applied in the future to determine which significant inputs could be used for training the NN.

### Adaptive Learning Rate

The adaptive learning rate H2O uses for its gradient descent algorithm was tweaked by manipulating two of its factors: Rho, the adaptive learning rate time decay factor, and Epsilon, the adaptive learning rate time smoothing factor. Rho relates to memorizing past weight updates and affects the influence of past gradients. Epsilon assists the model with encountering and overcoming local minima to find a global minimum more successfully. In our tests with Rho, we trained and tested the same neural model configuration with Rho values ranging from 0.01 to 0.99 and incrementing by 0.01 each time. From these tests we recorded the AUC, Accuracy, Precision, Recall, Precision-Recall AUC, and F1 score. The tests showed that the model performed logarithmically better and peaked at 0.99, which is also the default value used by the H2O Deep Learning model. The same tests were conducted with the Epsilon parameter, but this time the values ranged from 1E-10 to 1E-5 and the power was incremented by 0.1 each time. Like the Rho tests, the model performance improved logarithmically until an approximate Epsilon value of 5E-7. The default value for Epsilon is 1E-8. These tests were also conducted for the L1 Regularization

metric to reduce the possibility of overfitting while still maintaining adequate model performance regarding the same recorded variables from the above tests. With a default value of  $1E-5$ , tests were conducted from  $1E-6$  to  $1E-4$  with L1 being incremented by 0.1 each time. These tests showed little to no improvement in the model's performance, and the conclusion is that it did not affect model performance in relation to our data.

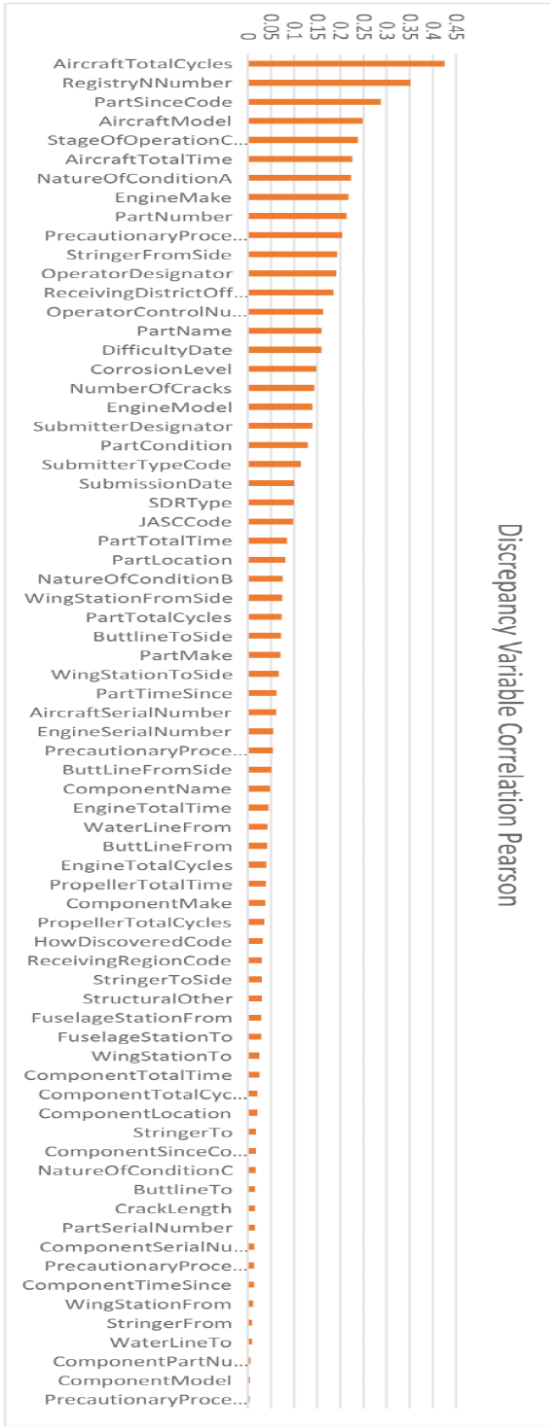


Fig. 14: Pearson Correlation Bar Chart Between All Inputs and Discrepancy

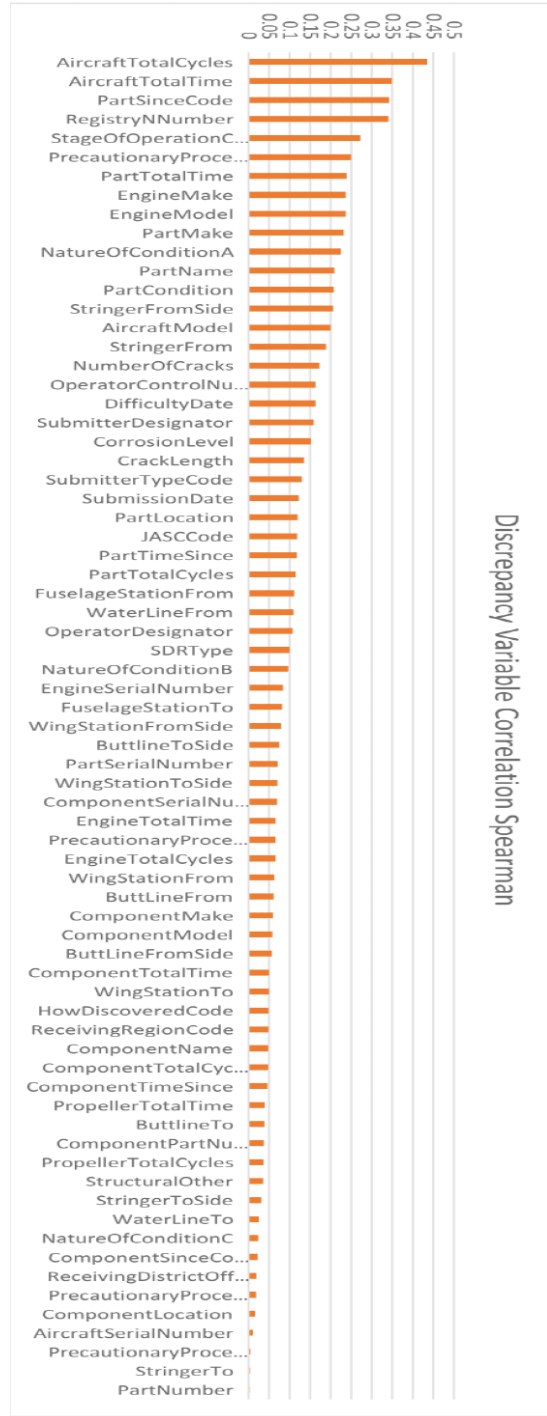


Fig. 15: Spearman Correlation Bar Chart Between All Inputs and Discrepancy

## **CONCLUSION**

In this thesis we propose a hybrid learning strategy by integrating a neural network with decision tree. The hybrid algorithm is tested with the Federal Aviation Administration (FAA) data for Boeing 737. Several simulated experiments have been performed to test the efficacy of the proposed hybrid method. The method is tested with various network architectures, activation functions, and different hidden layers. The hybrid method is also verified by selecting the contributing input features, and the similar prediction results confirm that it successfully identified the redundant features.

To optimize our NN, four primary tests were conducted with regard to classifying Discrepancy; 1) the hidden layer architecture to classification performance, 2) the total number inputs and of significant inputs to classification accuracy, 3) the performance of a hybrid DT – NN model to the typical NN model, and 4) the correlation of variables to all other variables. The first test demonstrated that performance peaks when the number of hidden layers is around 3 and steadily drops off until the model becomes unstable and is no better than guessing. The second test showed that the number of inputs could be reduced from 72 to 11 significant inputs without a reduction in accuracy of the NN. The test further validated this by showing that AUC, Accuracy, Recall and Precision stabilize with the 11 significant inputs and gradually deteriorate with the addition of new inputs. The third test shows that the hybrid DT – NN model outperforms a standalone NN model by comparing Precision vs. Recall. Finally, the fourth test further showed via heatmaps that only a small number of inputs have a high correlation with Discrepancy. Using these four tests, we show that the significant input features can be identified and that the total number of features can be reduced without affecting the accuracy of the NN. The hybrid learning method can be tested in more case studies in the future.



## REFERENCES

- [1] A. H. Vo, T. R. Van Vleet, R. R. Gupta, M. J. Liguori, and M. S. Rao, "An overview of machine learning and big data for drug toxicity evaluation," *Chemical Research in Toxicology*, vol. 33, no. 1, pp. 20–37, 2019.
- [2] F. Samie, L. Bauer, and J. Henkel, "From cloud down to things: An overview of machine learning in internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4921–4934, 2019.
- [3] I. H. Sarker, A. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big Data*, vol. 7, no. 1, pp. 1–29, 2020.
- [4] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Information Sciences*, vol. 177, no. 18, pp. 3799–3821, 2007.
- [5] R. R. Bies, M. F. Muldoon, B. G. Pollock, S. Manuck, G. Smith, and M. E. Sale, "A genetic algorithm-based, hybrid machine learning approach to model selection," *Journal of Pharmacokinetics and Pharmacodynamics*, vol. 33, no. 2, pp. 195–221, 2006.
- [6] S. Mohan, C. Thirumalai, and G. Srivastava, "Effective heart disease prediction using hybrid machine learning techniques," *IEEE Access*, vol. 7, pp. 81542–81554, 2019.
- [7] C. Qi, H.-B. Ly, Q. Chen, T.-T. Le, V. M. Le, and B. T. Pham, "Flocculation-dewatering prediction of fine mineral tailings using a hybrid machine learning approach," *Chemosphere*, vol. 244, pp. 125450, 2020.
- [8] Kinnison, H. A., Siddiqui, T., "Aviation Maintenance Management," *McGraw-Hill Professional*, New York, New York, United States, 2012.
- [9] Marx, D., A., Curtis R., G., "Human error in aircraft maintenance." *Aviation psychology in practice*, pp. 87-104, 1994.
- [10] Wang, T., Lu-Han, C., "Psychological and physiological fatigue variation and fatigue factors in aircraft line maintenance crews." *International Journal of Industrial Ergonomics* 44, vol. 1, pp. 107-113, 2014.
- [11] Segev, A., "Adaptive Ontology Use for Crisis Knowledge Representation," *Int. J. of Information Systems for Crisis Response and Management*, 1(2), pp. 16-30, April-June 2009.
- [12] Segev, A., Jihan, S. H., "Context Ontology for Humanitarian Assistance in Crisis Response," *Proceedings of the 10th International ISCRAM Conference*- T. Comes, F. Fiedrich, S. Fortier, J. Geldermann and T. Muller, eds. Baden-Baden, Germany, May 2013.

- [13] Sriram, C., Haghani, A., “An optimization model for aircraft maintenance scheduling and re-assignment,” *Transportation Research Part A: Policy and Practice*, vol. 37(1), pp. 29-48, 2003.
- [14] McDonald, N., Corrigan, S., Daly, C., Cromie, S., “Safety management systems and safety culture in aircraft maintenance organizations,” *Safety Science*, vol. 34(1), pp. 151-176, 2000.
- [15] Endsley, M. R., Robertson, M. M., “Situation awareness in aircraft maintenance teams,” *International Journal of Industrial Ergonomics*, vol. 26(2), pp. 301-325, 2000.
- [16] Vianna, W., O., L., Yoneyama, T., “Predictive maintenance optimization for aircraft redundant systems subjected to multiple wear profiles.” *IEEE Systems Journal* 12, no. 2: 1170-1181. 2017.
- [17] Hobbs, A., Williamson, A., “Associations between errors and contributing factors in aircraft maintenance.” *Human factors* 45, no. 2: 186-201, 2003.
- [18] Papakostas, N., Papachatzakis, P., Xanthakis, V., Mourtzis, D., Chryssolouris, G., “An approach to operational aircraft maintenance planning,” *Decision Support Systems*, Vol. 48(4), pp. 604-612, 2010.
- [19] De Crescenzo, F., Fantini, M., Persiani, F., Di Stefano, L., Azzari P., Salti, S., “Augmented reality for aircraft maintenance training and operations support,” *IEEE Computer Graphics and Applications*, vol. 31(1), pp. 96-101, 2011.
- [20] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, “A survey of machine learning for big data processing,” *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 67, 2016.
- [21] M. A. Fattah, “A hybrid machine learning model for multi-document summarization,” *Applied Intelligence*, vol. 40, no. 4, pp. 592–600, 2014.
- [22] K. Polat and S. Gunes, “A novel hybrid intelligent method based on c4.5 decision tree classifier and one-against-all approach for multiclass classification problems,” *Expert Systems with Applications*, vol. 36, no. 2, pp. 1587–1592, 2009.
- [23] H. Lu and X. Ma, “Hybrid decision tree-based machine learning models for short-term water quality prediction,” *Chemosphere*, vol. 249, p. 126169, 2020.
- [24] Z. Arabasadi, R. Alizadehsani, M. Roshanzamir, H. Moosaei, and A. A. Yarifard, “Computer aided decision making for heart disease detection using hybrid neural network-genetic algorithm,” *Computer methods and programs in biomedicine*, vol. 141, pp. 19–26, 2017.
- [25] B. T. Pham, D. T. Bui, I. Prakash, and M. Dholakia, “Hybrid integration of multilayer perceptron neural networks and machine learning ensembles for landslide susceptibility assessment at Himalayan area (India) using gis,” *Catena*, vol. 149, pp. 52–63, 2017.

- [26] W. Chen, S. Chen, H. Zhang, and T. Wu, "A hybrid prediction model for type 2 diabetes using k-means and decision tree," in *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pp. 386–390, IEEE, 2017.
- [27] C.-C. Lin, L. Shu, D.-J. Deng, T.-L. Yeh, Y.-H. Chen, and H.-L. Hsieh, "A MapReduce-based ensemble learning method with multiple classifier types and diversity for condition-based maintenance with concept drifts," *IEEE Cloud Computing*, vol. 4, no. 6, pp. 38–48, 2017.
- [28] I. H. Laradji, M. Alshayeb, and L. Ghouti, "Software defect prediction using ensemble learning on selected features," *Information and Software Technology*, vol. 58, pp. 388–402, 2015.
- [29] X. Yang, D. Lo, X. Xia, and J. Sun, "Tlel: A two-layer ensemble learning approach for just-in-time defect prediction," *Information and Software Technology*, vol. 87, pp. 206–220, 2017.
- [30] Z. Li, K. Goebel, and D. Wu, "Degradation modeling and remaining useful life prediction of aircraft engines using ensemble learning," *Journal of Engineering for Gas Turbines and Power*, vol. 141, no. 4, 2019.
- [31] J. Li, X. Li, and D. He, "A directed acyclic graph network combined with cnn and lstm for remaining useful life prediction," *IEEE Access*, vol. 7, pp. 75464–75475, 2019.
- [32] Sethi, I. K., "Entropy nets: from decision trees to neural networks," *Proceedings of the IEEE*, vol. 78(10), pp. 1605-1613, 1990.
- [33] Roe, B. P., Yang, H.-J., Zhu, J., Liu, Y., Stancu, I., McGregor, G., "Boosted decision trees as an alternative to artificial neural networks for particle identification," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 543(2-3), pp. 577-584, 2005.
- [34] Tso, G. K. F., Yau, K. K. W., "Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks," *Energy*, vol. 32(9), pp. 1761-1768, 2007.
- [35] Pal, P., Datta, R., Segev, A., "A neural net-based prediction of sound pressure level for the design of Aerofoil," *Proceedings of Fuzzy and Neural Computing Conference (FANCCO)*, 2019.
- [36] Xhemali, D., Hinde, C. J., Stone, R. G., "Naive bayes vs. decision trees vs. neural networks in the classification of training web pages," *International Journal of Computer Science Issues*, 4 (1), pp. 16-23, 2009.
- [37] Pal, P., Datta, R., Segev, A., Yasinsac, A., "Condition Based Maintenance of Turbine and Compressor of a CODLAG Naval Propulsion System using Deep Neural Network." *6<sup>th</sup> International Conference on Artificial Intelligence and Applications (AIAP-2019)*, 2019.

- [38] Pal, P., Datta, R., Rajbansi, D., Segev, A., “A Neural Net Based Prediction of Sound Pressure Level for the Design of the Aerofoil.” *In Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, pp. 105-112. Springer, Cham, 2019.
- [39] Segev, A., Datta, R., Benton, R., Curtis, D., “OINNIONN: outward inward neural network and inward outward neural network evolution.” *In Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 79-80. 2019.
- [40] Curram, S. P., Mingers, J., “Neural networks, decision tree induction and discriminant analysis: an empirical comparison,” *Journal of the Operational Research Society*, vol. 45(4), pp. 440-450, 1994.
- [41] West, D., “Neural network credit scoring models,” *Computers & Operations Research*, vol. 27(11-12), pp. 1131-1152, 2000.
- [42] Renato, D., M., Nascimento, C., L., “Prognostics of aircraft bleed valves using a SVM classification algorithm.” *In 2012 IEEE Aerospace Conference*, pp. 1-8. IEEE, 2012.
- [43] Bonnie Lida, R., Hamblin, C., J., Chaparro, A., “Classification and analysis of errors reported in aircraft maintenance manuals.” *International Journal of Applied Aviation Studies* 8, no. 2: 295, 2008.
- [44] Quinlan, J. R., “Induction of decision trees,” *Machine Learning*, vol. 1(1), pp. 81-106, 1986.
- [45] J. Carson, K. Hollingsworth, R. Datta, and A. Segev, “Failing & falling (f&!f): Learning to Classify Accidents and Incidents in Aircraft Data,” *2019 IEEE International Conference on Big Data (Big Data)*, pp. 4357 – 4365, IEEE, 2019.
- [46] J. Carson, K. Hollingsworth, R. Datta, G. Clark, and A. Segev, “A Hybrid Decision Tree – Neural Network (DT – NN) Model for Large-scale Classification Problems,” *2020 IEEE International Conference on Big Data (Big Data)*, pp. 4103 – 4111, IEEE, 2020.
- [47] J. Carson, K. Hollingsworth, R. Datta, and A. Segev; “Failing and Not Falling (F&!F): Data-Enabled Classification Learning of Aircraft Accidents and Incidents,” *Data-enabled Discovery and Applications*, vol. 4, 9, 2020.