International Journal of Innovative Research and Scientific Studies, 8(12) 2025, pages: 400-412

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

Data-driven torque identification of turboprop engines using optimized feedforward neural networks

Nguyen Khanh Huyen[1*], Nguyen Thi Anh[1], Nguyen Tien Dat[1], Nguyen Hai Duong[1]

[1]*Viettel High Technology Industries Corporation, Viettel Group, Hanoi, Vietnam.*

Corresponding author: Nguyen Khanh Huyen (*Email: huyennk3@viettel.com.vn*)

Abstract

This paper presents a research methodology for identifying the Pratt & Whitney Canada PW127G turboprop engine from simulation data using optimized feedforward neural networks (FNN). A set of measurable variables - ground speed, throttle lever angle, pressure altitude h_p, high-pressure spool speed N_H, and propeller speed N_P - is used to predict normalized engine torque, providing a surrogate engine model suitable for integration into flight simulators. The methodology follows a two-stage strategy. First, a baseline L-BFGS–trained FNN is combined with two architecture-search methods, Extended Great Deluge (EGD) and Bayesian Optimization (BO). On the turboprop dataset, BO achieves a lower test RMSE than EGD and is therefore selected as the preferred architecture optimization strategy. Second, BO is fixed and used to optimize two FNN configurations: Baseline FNN with inputs (ground speed, throttle lever angle, pressure altitude h_p, propeller speed N_P) and Core-Enhanced FNN additionally including high-pressure spool speed N_H. The optimized Core-Enhanced FNN significantly reduces the root mean square error from 1.066 to 0.4834 on testing data, corresponding to an average error reduction of about 55% compared with Baseline FNN, and also decreases mean relative error and error variance, confirming the importance of core-speed information for high-fidelity torque prediction. The results demonstrate that L-BFGS–trained FNNs, combined with BO-based architecture search and simulation-derived data, provide an effective and computationally efficient surrogate engine model for turboprop torque (and indirectly thrust) estimation in advanced flight simulation and training applications.

Keywords: Aircraft system identification, Artificial neural networks, Bayesian Optimization, Engine torque, Extended Great Deluge algorithm, Feedforward neural network, Flight simulation, Turboprop engines.

DOI: 10.53894/ijirss.v8i12.11110
Funding: This study received no specific financial support.
History: Received: 7 November 2025 / **Revised:** 10 December 2025 / **Accepted:** 15 December 2025 / **Published:** 24 December 2025

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Acknowledgments: The authors sincerely appreciate the support and valuable resources provided by the Modeling and Simulation Center, a division of Viettel High Technology Industries Corporation, Viettel Group, during this study. Additionally, we deeply acknowledge the leadership team, and team members at the Modeling and Simulation Center for their dedicated assistance, insightful feedback, and significant contributions to the testing and refinement of the solution examined in this project.
Publisher: Innovative Research Publishing

1. Introduction

The excellent propulsive efficiency of turboprop engines at low to medium subsonic speeds makes them popular in both military and regional aircraft. Accurate identification of thrust- and torque-related characteristics in turboprop engines is essential for model-based control, health management, and aircraft performance monitoring, as well as for the development and validation of high-fidelity, regulatory-standard flight simulators and propulsion digital twins. In particular, realistic flight simulators and mission-analysis tools require engine models that can reproduce thrust and torque responses across a wide operating envelope; otherwise, the simulated aircraft performance may significantly deviate from real-world behavior.

Temperature, pressure, power, thrust, and torque have traditionally been calculated using mathematical models of the multi-stage gas path, which includes the compressor, shaft, combustor, turbine, and nozzle [1-3]. These analytical, physics-based models usually rely on simplifying assumptions (e.g., constant operating conditions, linearized component maps), even though they offer physically interpretable descriptions. In reality, the accuracy and adaptability of the thermodynamic and mechanical subsystems under varying flight conditions is limited by their strong coupling and nonlinear interactions.

An increasing amount of research has proposed data-driven modeling techniques to overcome these constraints, supported by advances in computational capability. Neural networks, for instance, are frequently used to forecast thrust or fuel consumption from flight or simulation data. Yildirim Dalkiran and Toraman [4] developed a multilayer perceptron (MLP) model to forecast net thrust using altitude, Mach number, and atmospheric characteristics as inputs. Zaag, et al. [5] proposed an improved MLP-based identification framework for AE3007 turbofan engines during cruise flight. In the same way, more intricate models for the dynamic correction of predicted engine parameters have been developed using Elman–AdaBoost neural networks and LSTM architectures. Sabzehali, et al. [6] investigated the use of artificial neural networks (ANNs) to forecast fuel consumption and energy characteristics of the PW100 turbofan engine. Vladov, et al. [7] modeled turboshaft engines using NARX neural networks trained with real data. Andrianantara, et al. [8] use ANNs to model the performance of the CF34-8C5B1 turbofan engine for the CRJ-700 aircraft. Chati and Balakrishnan [9] applies machine learning techniques, including CART and Least Squares Boosting, to model the fuel flow rate of aircraft engines based on flight recorder data.

More recently, several works have focused specifically on turboprop engines or closely related configurations. Kayaalp and Metlek [10] used a feedforward ANN to predict burning performance and emission indices (CO, NOx, unburned HC) of a turboprop motor as functions of air–fuel ratio, shaft speed, and fuel flow; their model achieved very low prediction errors and demonstrated that ANN-based surrogates can replace time-consuming combustion simulations for design and optimization. Dursun, et al. [11] compared different machine-learning approaches-including MLP, random forest, and LSTM—for modeling performance and thermodynamic metrics of a conceptual turboprop engine across a wide operating envelope, and reported that deep/recurrent architectures can capture nonlinear engine behavior with high accuracy (MAPE on key metrics typically below 3–5%). Psaropoulos, et al. [12] developed an ANN-based surrogate model for turboprop engine performance using data generated by a detailed thermodynamic cycle tool; an MLP trained on a Latin-hypercube design was able to predict shaft power and fuel flow with mean absolute percentage errors of about 3.3% and 7%, respectively, enabling fast exploration of engine designs and full flight envelopes. Earlier turboprop-focused work by Baklacioglu, et al. [13] combined a genetic algorithm with ANN training to dynamically model the exergy efficiency of turboprop engine components, illustrating that hybrid GA–ANN strategies can be used both to optimize network topology and to improve convergence.

While extensive research has been conducted on turbofan and turbojet engines—particularly leveraging artificial neural networks (ANNs) and other machine-learning techniques for thrust prediction, performance estimation, and engine health monitoring, and for supplying realistic engine modules to flight simulators and propulsion digital twins—the application of such intelligent methods to turboprop engines remains relatively limited in the open literature. For turboprop configurations, existing studies primarily focus on thermodynamic optimization, emission analysis, or fault diagnostics, with relatively few efforts dedicated to real-time thrust and torque identification under varying flight regimes or to

providing sufficiently accurate torque/thrust models for use in training and certification-grade simulators. This gap is particularly important since turboprop engines, unlike turbofans, derive the majority of their thrust from the shaft-driven propeller, making conventional jet-based estimation techniques inadequate.

In this study, the main objective is to develop optimized artificial neural network (ANN) models for turboprop engine performance prediction. The proposed approach combines a feedforward neural network structure with two architecture–search strategies, namely Extended Great Deluge (EGD) and Bayesian Optimization (BO), in order to automatically tune the number of layers and neurons and thus improve generalization. To investigate the influence of input parameter selection, two ANN models are designed and compared: the first model uses speed, throttle lever angle, pressure altitude h_p, and propeller rotational speed N_P as inputs, whereas the second model augments this set with the high-pressure spool speed N_H. The comparative analysis of these two optimized models provides insight into both the effectiveness of the optimization algorithms (EGD vs BO) and the impact of including additional engine-state information on turboprop performance prediction accuracy.

2. Aircraft Description and Engine Mathematical Model

According to Pratt & Whitney Canada, the PW127G is a member of the PW100 turboprop engine family, which is designed for regional, utility, and tactical transport applications. This is also confirmed by Airbus Defence and Space, which list the PW127G as a PW100-series turboprop engine used on the C-295 aircraft [14]. It features a two-shaft configuration comprising a three-stage axial low-pressure compressor, a centrifugal high-pressure compressor, a single-stage high-pressure turbine, and a free turbine driving the propeller.

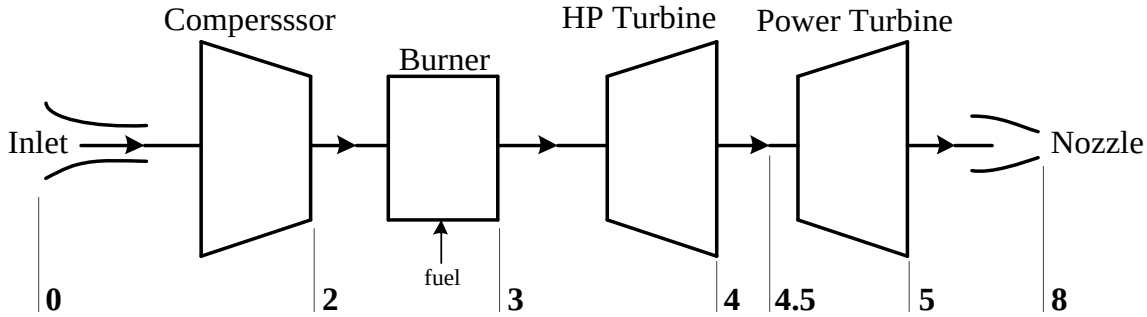

Figure 1.
Schematic illustration of the turboprop engine thermodynamic cycle.

In a turboprop engine (as shown in Figure 1), the incoming air flows through several distinct thermodynamic stages before it is converted into mechanical power. The process begins as ambient air enters through an inlet diffuser and flows into the compressor, where it compression to reaching the pressure level required for efficient combustion. This high-pressure air enters the combustor, where it is mixed with fuel and burned. These gases first pass through the high-pressure turbine (HPT), which extracts energy to drive the compressor via a concentric shaft. The partially expanded gas then enters the power turbine, which is mechanically decoupled from the gas generator and is instead connected to the propeller via a reduction gearbox. The power turbine converts thermal energy into mechanical power, which is used to rotate the six-blade propeller. the majority of the thrust in a turboprop engine (typically 85–90%) is generated by the propeller**.** The residual high-velocity exhaust gas leaving the LPT contributes minimally to net thrust (usually <15%) [1, 2].

The primary output variable targeted in this study is the Torque Q, which is physically related to the shaft power P and the rotational speed N_p by the classical mechanical relationship

$$P = Q.2\pi N_P \quad (1)$$

This equation implies that any variation in either shaft power or shaft speed will directly affect the torque output. Thus, to model torque accurately, one must examine the external factors that influence both these components.

The shaft power P can be estimated thermodynamically by evaluating the enthalpy drop across the power (free) turbine [1].

$$P = \dot{m}_{4.5} c_P (T_{t4.5} - T_{t5}) \quad (2)$$

In which, $\dot{m}_{4.5}$ is Mass flow rate between HPT and power turbine, c_P is Pressure coefficient, $T_{t4.5}$ is Temperature between HPT and T_{t5} is power turbine and Temperature after power turbine.

Each components of the shaft power equation are influenced by the inlet conditions. The inlet pressure and temperature are functions of altitude (h_p) and speed of aircraft (V). At the combustion chamber, the amount of fuel, which is controlled by throttle (TLA), injected plays a critical role in determining the total thermal energy input to the system. This variation in fuel supply directly influences the amount of energy converted into shaft power. This calculation requires full knowledge of the engine's design parameters and how these parameters change across the thermodynamic cycle. However, these internal thermodynamic conditions are often not disclosed by the engine manufacturers, do not release detailed information about the thermodynamic conditions inside the engine. The HPT uses a portion of the energy to drive the compressor, so that add

the speed of HPT (N_H) a component effect to the shaft power to increase the accuracy and robustness of turboprop torque and thrust prediction.

$$Q = f(V, TLA, h_p, N_H, N_p) \quad (3)$$

3. Methodology and Model Identification

3.1. Engine Performance Database

The data set used in this study is generated from a high-fidelity turboprop engine and aircraft performance simulation environment representative of a platform equipped with PW127G engines. The simulator provides time histories of both engine and aircraft states over complete mission profiles, including climb, cruise and descent segments. In total, 36,587 samples were extracted under a wide range of throttle schedules and operating conditions, ensuring representative coverage of engine behaviour during acceleration, deceleration, and quasi–steady-state regimes across the full flight envelope.

The selected features comprise ground speed, throttle lever angle, pressure altitude, high-pressure spool speed, propeller speed, and engine torque as shown in Table 1**.** These parameters were chosen because they are directly measurable in typical avionics and engine-monitoring systems and exhibit strong physical correlation with the turboprop thrust-generation mechanism, making them suitable inputs and outputs for data-driven engine modelling and flight-simulation applications.

Table 1.
Summary of Input and Output Variables Used for the Turboprop Engine Model.

Variable	Symbol	Unit	Min.	Max.	Description	Type
Throttle Lever Angle	TLA	degree	3.91	80.64	Pilot's commanded power level	Input
Ground Speed	V	knots	0	280	Aircraft longitudinal velocity	Input
Pressure Altitude	h_p	ft	98	17,023	Flight altitude based on static pressure	Input
High-Pressure Spool Speed	N_H	%	65.63	100	Rotational speed of high-pressure turbine	Input
Propeller Speed	N_p	%	24.88	100	Rotational speed of the propeller shaft	Input
Engine Torque	Q	%	0.06	100	Output torque from the power turbine	Output

Before network training, data preprocessing was conducted to ensure numerical stability and improve learning convergence. Missing values (NaN) and outliers were removed using statistical filtering.

Because of the computational characteristics of neural networks, directly inputting raw data with large magnitudes can cause instability and hinder convergence. To mitigate this**,** input features were normalized using Z-score normalization, the normalization equations are expressed as:

$$x_{norm} = \frac{x - \mu}{\sigma} \quad (1)$$

where x represent the original input, respectively, and $\mu,\ \sigma$ denote the mean and standard deviation of the input features.

3.2. Neural Network–Based Engine Model

The proposed method aims to identify the nonlinear mapping between the measurable engine variables and the output engine torque of a turboprop propulsion system. A Feedforward Neural Network (FNN) is employed as a universal nonlinear approximator to capture the complex input–output relationships of the engine [15-17].

The overall training framework integrates three main components:

1. FNN-based modeling of the engine dynamics;
2. Backpropagation-based learning using the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) optimizer; and
3. A hybrid optimization strategy combining the Extended Great Deluge (EGD) algorithm and Bayesian Optimization (BO) for enhanced convergence and generalization.

After normalization and data partitioning, the FNN model is trained to minimize the mean squared error (MSE) between the predicted and measured torque values. EGD is applied to refine the weight initialization and assist in escaping local minima during training, while BO is employed to determine optimal hyperparameters for network architecture and learning process.

3.2.1. FNN-Based Modeling and Training

The primary objective of this study is to identify the nonlinear functional relationship between a set of measurable engine parameters $x = [x_1, x_2, \dots, x_n]^T$ and the output engine torque T_e of a turboprop propulsion system. This relationship can be generally expressed as:

$$T_e = f(x) + \epsilon \quad (5)$$

Where $f(.)$ denotes the unknown nonlinear mapping to be identified, and ϵ represents the modeling error or process noise. A FNN is employed as a universal approximator capable of representing $f(.)$ with arbitrary accuracy given sufficient hidden neurons [1, 2]. The FNN structure consists of K layers with parameter W^l and b^l denoting the weights and biases of the l^{th} layer. The network's forward propagation can be described recursively as:

$$h^l = \phi^l(W^l h^{l-1} + b^l), \quad l = 1,2,\dots,L \tag{6}$$

With $h^0 = x$ being the input layer, and $\phi^l(.)$ denoting the activation function. The final output is given by:

$$\widehat{T_e} = W^L h^{L-1} + b^L \tag{7}$$

The model parameters $\theta = \{W^l, b^l\}_{l=1}^{L}$ are optimized by minimizing the Mean Squared Error (MSE) cost function:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(T_{e,i} - \widehat{T_{e,i}}\right)^2, \tag{8}$$

where N denotes the number of training samples.

We train the proposed feedforward neural network (FNN) using the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm, a quasi-Newton method that exploits curvature information with low memory cost [18].

At iteration k, the gradient is

$$g_k = \nabla_\theta J(\theta_k) \tag{9}$$

and the L-BFGS update is

$$\theta_{k+1} = \theta_k + \alpha_k p_k\,, \qquad p_k = -H_k^{-1} g_k \tag{10}$$

Where α_k is obtained by line search and H_k^{-1} is an implicit approximation of the inverse Hessian. Instead of storing H_k^{-1} explicitly, L-BFGS only keeps the last m pairs

$$s_k = \theta_{k+1} - \theta_k\,, y_k = g_{k+1} - g_k \tag{11}$$

Here, s_k represents the parameter step and y_k the corresponding change in gradient; together they encode local curvature through the secant relation $\nabla^2 J(\theta_k) \approx y_k$ [3].

The search direction p_k is then computed via the standard two-loop recursion using these (s_j, y_j) pairs, with complexity $O(mn)$ instead of $O(n^2)$.

Empirical studies have shown that combining L-BFGS with neural networks can yield faster convergence and better generalization compared to purely first-order methods [19] which is particularly beneficial for high-fidelity system identification of aero-engines.

Building on this framework, the proposed neural network–based engine model is constructed to map a vector of measurable flight and engine parameters to the shaft torque output.

To investigate the impact of input selection, two alternative network configurations are considered. Baseline FNN (as shown in Figure 2) uses only V, TLA, h_p, and N_P as inputs, representing quantities typically available to conventional performance models and thrust-estimation methods. Core-Enhanced FNN (as shown in Figure 3) augments this set by including the core-speed variable N_H, which carries additional information about the gas-generator energy state and has been shown to correlate strongly with available power in turboprop engines. The comparative analysis of these two FNN configurations enables a quantitative assessment of how incorporating N_H improves torque prediction accuracy and robustness across the turboprop operating envelope.

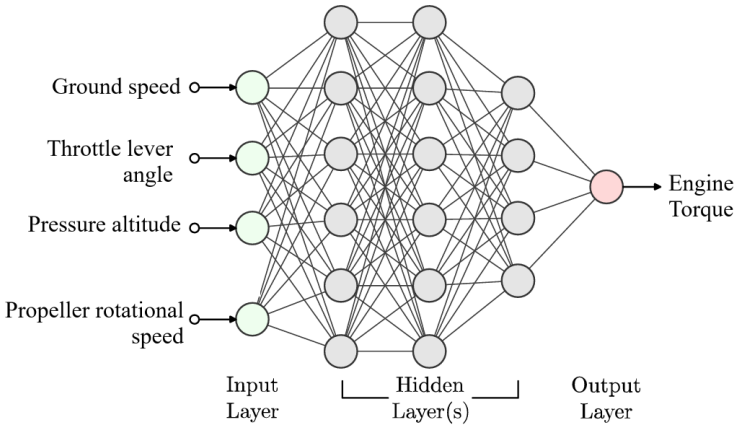

Figure 2.
Baseline FNN using four input parameters for turboprop engine torque prediction.

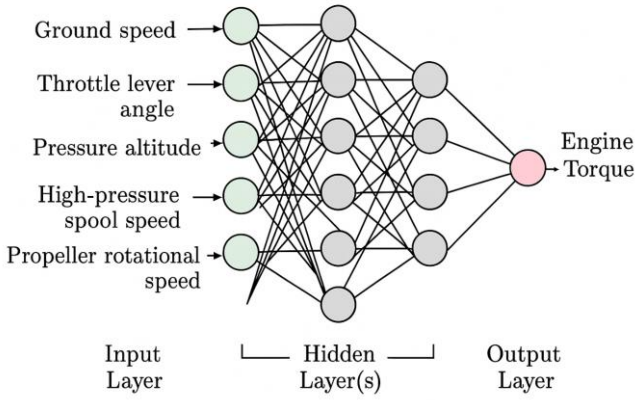

Figure 3.
Core-Enhanced FNN including high-pressure spool speed N_H for improved torque prediction.

3.2.2. Architecture Optimization: Extended Great Deluge vs Bayesian Optimization

The FNN described in Section 3.2.1 consists of L layers with weights and biases $\theta = \{W^l, b^l\}_{l=1}^{L}$
For a given architecture we specify:

- the number of hidden layers L_h,
- the number of neurons in each hidden layer $n_1, \dots, n_{L_h}$.

We encode the network architecture by the integer vector

$$a = \left(L_h, n_1, \dots, n_{L_h}\right) \in \Omega \quad (12)$$

Where Ω denotes the discrete architecture search space subject to bounds on L_h and n_ℓ.

Given an architecture X, the network parameters $\theta^*(a)$ are obtained by minimizing the MSE cost function using the L-BFGS algorithm:

$$\theta^*(X) = \text{argmin}_\theta\ J(\theta; a) \quad (13)$$

Where $J(\theta; a)$ is the training loss of the FNN with architecture a.
To compare different architectures, we evaluate the validation MSE

$$E(a) = J(\theta(X); a) \quad (14)$$

computed on a validation subset disjoint from the training data. The architecture optimization problem can then be written as:

$$a^* = argmin_{a\in\Omega}\, E(a) \quad (15)$$

The following subsections describe two alternative strategies used in this paper to solve this discrete optimization problem: Extended Great Deluge (EGD) and Bayesian Optimization (BO). Both methods operate on the same architecture space Ω; for each candidate a, the network is trained with L-BFGS and evaluated via $E(a)$.

3.3. Extended Great Deluge Algorithm

In this work, the Extended Great Deluge (EGD) algorithm is used as a standalone metaheuristic to optimize the neural-network architecture, and its performance is later compared against Bayesian Optimization. The objective is to determine the number of hidden layers and the number of neurons per layer that minimize the validation error of the turboprop engine model. The use of Great Deluge–type algorithms for discrete architecture search is motivated by their simplicity and proven effectiveness in combinatorial optimization and learning tasks, including timetabling, reliability, fuzzy cognitive map training, and neural-network-based aerodynamic and engine modeling [20, 21].

In particular, several studies have successfully coupled Extended Great Deluge with neural networks for aerodynamic coefficient prediction, morphing-wing actuator displacement prediction, and turbofan/turbofan-like engine modeling, reporting low prediction errors and robust generalization [5, 22, 23].

3.4. Initialization

We define the following EGD control parameters:

- ΔB: EGD step size (water-level decrement)
- MI: maximum number of EGD iterations per layer
- MN: maximum number of neurons per layer,
- ML: maximum number of hidden layers.

The search starts from a simple architecture with:

- Number of neurons $n = 1$.
- Number of layers $l = 1$

For this initial configuration, a neural network is trained (using the same training procedure and optimizer as all other experiments), and its validation error is denoted:

$$e^* = E(a_0) \tag{16}$$

The initial water level is set to: $B = e^*$.

This provides a first feasible solution and an initial threshold for EGD.

3.5. EGD Iterations for a Fixed Number of Layers

For each fixed number of layers l (from 1 up to ML), the algorithm performs an EGD-based search over the number of neurons per layer.

At each EGD iteration:

1. Random architecture proposal

A candidate number of neurons n is selected uniformly at random in the interval

$$n \in [1, MN]$$

A feedforward network with l hidden layers and n neurons in each layer is constructed.

2. Network training and error evaluation

Train the FNN with architecture a'using L-BFGS, and compute the validation error:

$$e = E(a') \tag{17}$$

3. EGD acceptance test

The candidate architecture is accepted if it is either better than the current best solution or lies below the current water level:

$$\mathrm{e} \leq \mathrm{e}^* \text{ or } \mathrm{e} \leq \mathrm{B}$$

- If accepted, the best error is updated: $e^* = e$, and the corresponding architecture configuration is stored.
- Otherwise, the candidate is rejected and the current best configuration remains unchanged.

4. Water-level update

After each iteration, the water level is decreased linearly: $B \leftarrow B - \nabla B$

This gradually tightens the acceptance condition, forcing the search to converge toward architectures with lower validation error [17].

These steps are repeated until the maximum number of iterations MI is reached for the current number of layers l.

3.6. Layer-wise Loop and Final Selection

After the EGD loop for a given l reaches ML, the number of layers is incremented:

$$l \leftarrow l + 1,$$

and the entire EGD process over neurons is repeated for this new depth, until $l > ML$. During this process, every accepted architecture and its corresponding error are recorded. At the end of the search, the network configuration with the lowest validation error across all explored depths and neuron counts is selected and saved as the EGD-optimized architecture.

3.7. Bayesian Optimization

In the second strategy, we apply Bayesian Optimization (BO) directly to the discrete architecture variable a. We again consider the architecture–performance mapping

$$f(a) = E(a) = J(\theta^*(a); a) \tag{18}$$

We model $f(a)$ as a black-box function over Ω and place a Gaussian Process (GP) prior on a suitable continuous embedding of the architecture vector a [24].

$$f(a) \sim GP(0, k(a, a'))$$

Given a set of previously evaluated architectures:

$$D_t = \left\{\left(a^i, f(a^i)\right)\right\}_{i=1}^{t} \tag{19}$$

the GP provides a posterior mean $\mu_t(a)$ and variance $\sigma_t^2(a)$ for each untried architecture. An acquisition function $\alpha(a)$ is then used to select the next architecture to evaluate.

In this work we employ the Expected Improvement acquisition function for minimization. Let

$$f_{min} = \min_{1 \leq i \leq t} f(a^i) \tag{20}$$

be the best validation error observed so far, and define

$$\gamma_t(a) = \frac{f_{min} - \mu_t(a)}{\sigma_t(a)} \tag{21}$$

when $\sigma_t(a) > 0$. Using the standard normal probability density function $\phi(.)$ and cumulative distribution function $\Phi(.)$,the Expected Improvement at architecture a is

$$EI_t(a) = \big(f_{min} - \mu_t(a)\big)\Phi\big(\gamma_t(a)\big) + \sigma_t(a)\phi\big(\gamma_t(a)\big) \quad (22)$$

and $\mathrm{EI}_t(a) = 0$ if $\sigma_t(a) = 0$.
At each BO iteration,

$$a_{t+1} = argmax_{a\in\Omega}\, EI_t(a) \quad (23)$$

is selected, the network with architecture a_{t+1} is trained using L-BFGS, and the new observation $f(a_{t+1})$ is added to D_t. This process is repeated until a pre-defined evaluation budget is exhausted. The architecture with the minimum observed validation error is returned as the BO-optimized design. This formulation closely follows the GP–EI framework proposed by Snoek, et al. [25] for hyperparameter optimization of neural networks

4. Simulation and Validation Result

This section presents the experiments conducted to evaluate the effectiveness of the proposed approach: Various optimization algorithms were applied to fine-tune the neural network hyperparameters (such as the number of layers, number of neurons, etc.) for the turboprop engine model identification problem; The study also investigates the influence of different parameters in constructing the neural network to determine the most efficient model configuration. Both models were trained using real aircraft data, with the training, validation, and testing datasets covering all flight phases — takeoff, climb, cruise, and descent **—** as described in Section III**.**

4.1. Criteria for Statistical Analysis

To quantitatively evaluate the performance of the identified turboprop engine models, three statistical indices were used: the Root Mean Square Error (RMSE)**,** the Mean Relative Error (MRE), and the Standard Deviation (STD) of the relative errors. These indicators collectively assess the prediction accuracy, generalization capability, and stability of the proposed neural network models.

The Root Mean Square Error (RMSE) is a widely used statistical measure that quantifies the average magnitude of prediction errors between the model outputs and the corresponding reference data. It is defined as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y_i})^2} \quad (24)$$

where N denotes the total number of data samples, y_i represents the measured value, and $\hat{y_i}$ is the corresponding predicted output obtained from the neural network.

A lower RMSE value indicates a more accurate fit of the model to the reference engine data, directly reflecting its ability to reproduce the nonlinear thermodynamic characteristics of the turboprop engine.
The relative error $\overline{\varepsilon_i}$ (in percentage) was also calculated as follows:

$$\overline{\varepsilon_i} = \frac{y_i - \hat{y_i}}{y_i} \quad (25)$$

The Mean Relative Error (MRE), denoted as μ, represents the average bias between the predicted and measured values, while the Standard Deviation (STD)**,** denoted as σ, measures the dispersion of these relative errors around their mean. They are given by:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}\overline{\varepsilon_i} \quad (26)$$

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\overline{\varepsilon_i} - \mu)^2} \quad (27)$$

By definition, the standard deviation σ represents the square root of the variance and reflects the degree of dispersion of the relative errors $\overline{\varepsilon_i}$ around their mean value μ.

For an ideal model, both the mean relative error μ and the standard deviation σ should be zero [17]. Consequently, the model is regarded as accurate when μ is close to zero and σ is below than 5%. This statistical criterion enables the assessment of model accuracy to be extended to the entire flight envelope of the aircraft and across all flight phases.

4.2. Hyperparameter Optimization and Algorithm Validation

To evaluate the effectiveness of the proposed optimization strategies, a comparative experiment was conducted between the Extended Great Deluge and Bayesian Optimization algorithms for hyperparameter tuning of the neural network model. Both methods were tested under identical data partitions and computational configurations, with the objective of minimizing the validation RMSE. This ensured that the comparison between Extended Great Deluge and Bayesian Optimization focused solely on the optimization strategy, without bias from hardware or data-related factors.

Table 2.
Performance comparison between Extended Great Deluge (EGD) and Bayesian Optimization (BO) for neural network hyperparameter tuning.

	Best RMSE train	RMSE test	Total times (s)	Configurations Tried
EGD	1.3179	1.4993	1184	22
BO	0.9989	1.066	565	30

Table 2 summarizes the results obtained from the two optimization methods. The EGD algorithm achieved a best validation RMSE of 1.3179 and a test RMSE of 1.4993, with a total optimization time of 1184 seconds across 22 configurations. In contrast, the BO method reached a superior best validation RMSE of 0.9989 and test RMSE of 1.066, while requiring only 565 seconds and exploring 30 configurations.

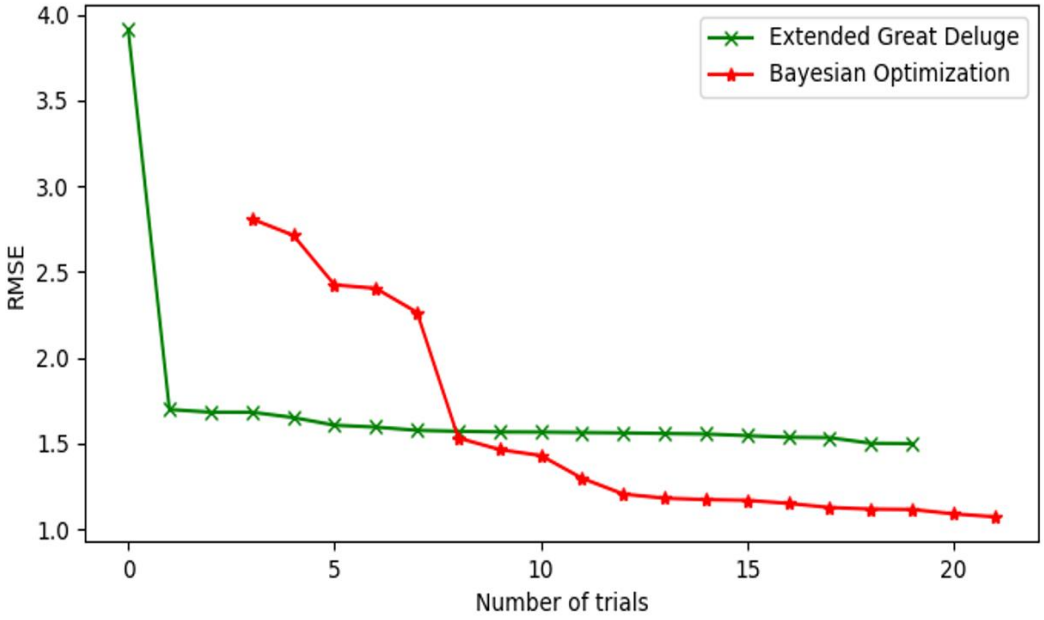

Figure 4.
RMSE comparison between Extended Great Deluge and Bayesian Optimization during hyperparameter tuning.

As illustrated in Figure 4, the RMSE of the BO approach decreases rapidly with the number of trials and stabilizes at a lower value compared to EGD. This indicates a faster convergence toward the optimal hyperparameter region.

4.3. Model Validation Across Engine Operating Conditions

After completing the hyperparameter optimization, the best configuration obtained from the Bayesian Optimization (BO) method—corresponding to the lowest validation RMSE—was selected for further evaluation. This optimized set of hyperparameters was then applied to both Baseline FNN and Core-Enhanced FNN to ensure a fair comparison between different input configurations.

Figure 5 illustrates the convergence behaviors of the two models trained with the L-BFGS optimizer. A model is considered converged when both the training loss and the gradient norm approach asymptotic stability, i.e., when the rate of change between successive iterations becomes negligible. Both models exhibit rapid convergence within the first 10 iterations, where the training loss and the gradient norm decrease sharply and subsequently stabilize near zero. The second model demonstrates a slightly faster convergence rate and smoother gradient decay, indicating better numerical stability and more efficient parameter adjustment. These results confirm that the training process successfully reached a stable local minimum with negligible gradient magnitude.

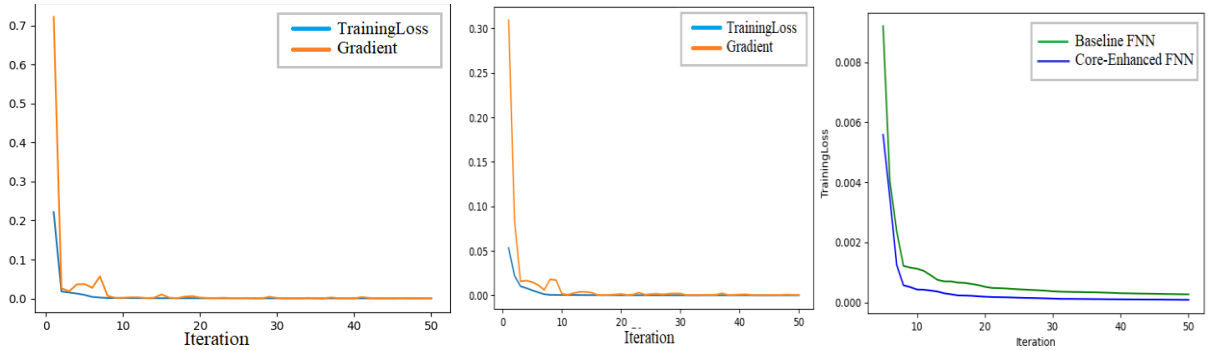

Figure 5.
Training loss and gradient norms of Baseline FNN and Core-Enhanced FNN throughout the training iterations.

Table 3.
Summarizes the statistical performance metrics obtained for the training and testing datasets.

	RMSE train	RMSE test	MRE	Standard Deviation
Baseline FNN	0.9989	1.066	0.0173	0.0781
Core-Enhanced FNN	0.4649	0.4834	0.0098	0.0202

Table 3 summarizes the results of training and testing for both models. Baseline FNN achieved an RMSE of 0.9989 during training and 1.066 on the test set, with a mean relative error (MRE) of 0.0173. In contrast, Core-Enhanced FNN demonstrated significantly improved performance, achieving RMSE values of 0.4649 (training) and 0.4834 (testing), while reducing the MRE to 0.0098. The standard deviation of the prediction error also decreased notably from 0.0781 to 0.0202, indicating higher stability and generalization capability.

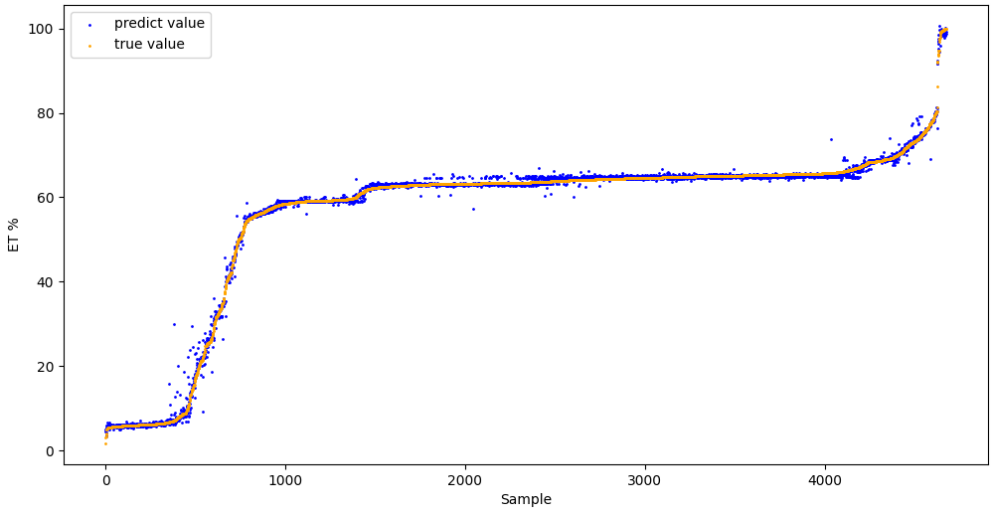

Figure 6.
Comparison between predicted and true torque values for Baseline FNN on the test dataset.

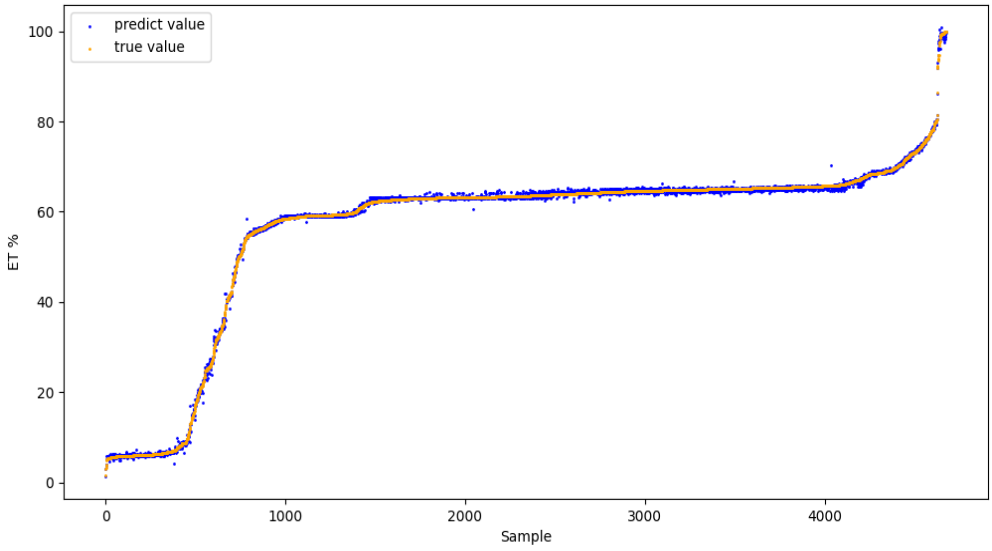

Figure 7.
Comparison between predicted and true torque values for Core-Enhanced FNN on the test dataset.

Figures 6 and 7 illustrate the comparison between the predicted and measured torque values on the test dataset. Baseline FNN exhibits visible deviations in low- and high-thrust regions and particularly during rapid acceleration or deceleration of the engine. Meanwhile, Core-Enhanced FNN shows excellent agreement with the experimental data across the entire operating range, including transient conditions. The inclusion of high-pressure spool speed allows the model to capture the dynamic coupling between the core spool and the propeller turbine, resulting in smoother and more accurate predictions when the engine power changes rapidly.

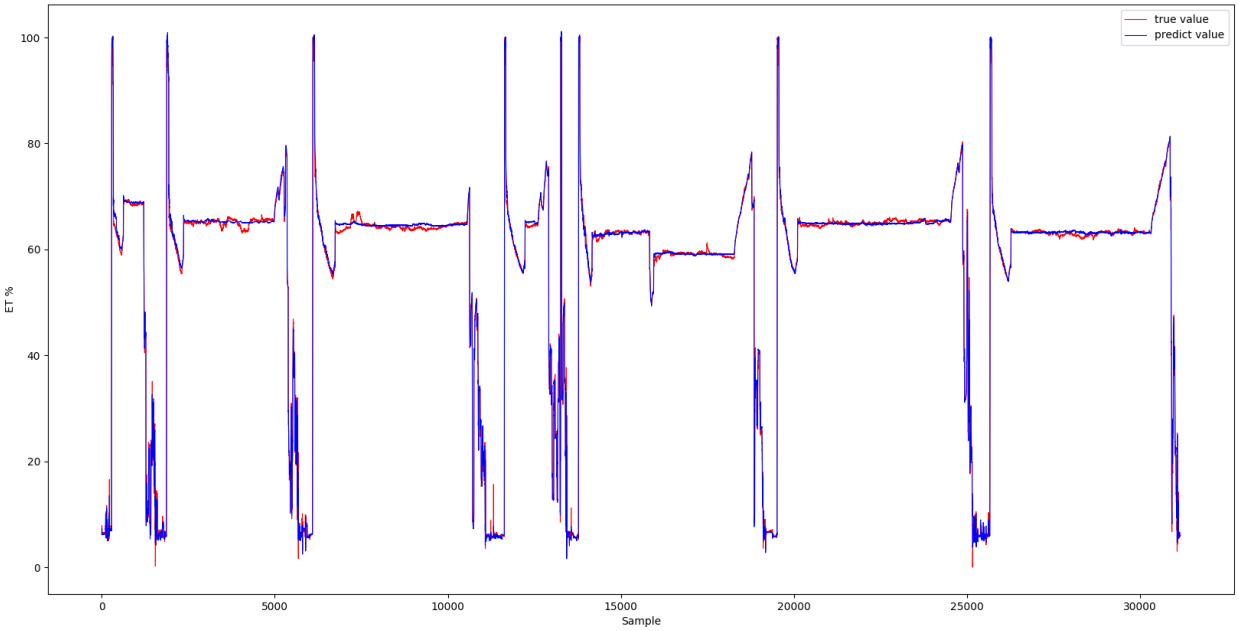

Figure 8.
Predicted versus true torque of Baseline FNN across the entire dataset.

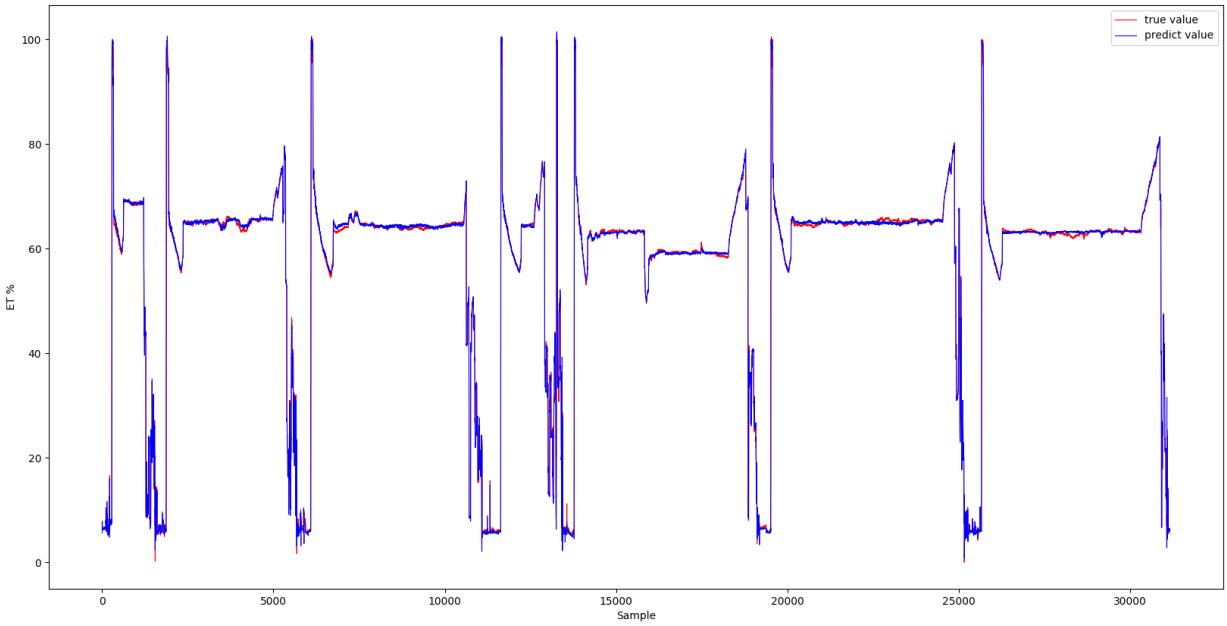

Figure 9.
Predicted versus true torque of Core-Enhanced FNN across the entire dataset.

To further assess dynamic behavior, Figures 8 and 9 show model predictions versus true torque across the entire dataset, including multiple acceleration and deceleration cycles. In Fig. 4, Baseline FNN fails to follow the true signal during rapid power changes, producing time-lagged and overshoot patterns, as seen in sample ranges such as 5000–6000, 6500–7000, 1100–1200, … Conversely, in Fig. 5, Core-Enhanced FNN tracks the true values closely throughout both steady-state and transient phases, demonstrating its improved responsiveness and robustness.

Overall, the validation results confirm that the inclusion of the high-pressure spool speed parameter significantly enhances the model's capability to represent transient engine behavior. Core-Enhanced FNN provides a more accurate, stable, and physically consistent estimation of turboprop torque compared to Baseline FNN, emphasizing the advantage of incorporating dynamic engine parameters in data-driven modeling.

5. Conclusion

This paper introduced a data-driven identification framework for turboprop engines utilizing feedforward neural networks and sophisticated architectural optimization. Driven by the limited availability of ANN-based thrust and torque models for turboprop engines in existing literature, this study concentrated on the aircraft, which is equipped with PW127G engines, utilizing flight-data-derived variables as inputs for an ANN torque estimator. A mathematical representation of the turboprop propulsion system is first introduced, highlighting the dominant role of the propeller in producing torque. Based

on this physical understanding and the available flight data, a set of measurable input parameters—ground speed, throttle lever angle, pressure altitude, high-pressure spool speed, and propeller speed—is established as the basis for predicting torque and thrust.

The methodological approach consists of a two-stage procedure. In the first stage, a baseline FNN-based torque model enhanced with the L-BFGS algorithm is used to evaluate two architectural optimization methods: Extended Great Deluge (EGD) and Bayesian Optimization (BO). The objective is to determine which search strategy is more effective for tuning the number of neurons and layers.The results show that BO consistently outperforms EGD on the considered turboprop data set: the best BO configuration reduced the test RMSE from 1.4993 (EGD) to 1.066 (≈29% improvement). This initial phase suggests BO as a more efficient and robust architecture-search strategy for the proposed FNN-based turboprop torque model.

In the second stage, BO was established as the selected optimization strategy, and the impact of input selection was examined by contrasting two FNN configurations: Baseline FNN, using (V, TLA, h_p, N_p) and Core-Enhanced FNN, which augments the input vector with the core-speed variable N_H. Both models were trained with L-BFGS and had their architectures tuned by BO. The optimized Baseline FNN achieved training and test RMSE values of 0.9989 and 1.066, a mean relative error (MRE) of 0.0173, and a standard deviation of 0.0781. When N_H was included (Core-Enhanced FNN), the training and test RMSE values decreased to 0.4649 and 0.4834 (≈50–55% reduction), the MRE dropped to 0.0098 (≈43% reduction), and the standard deviation of the error decreased to 0.0202 (more than 70% reduction). These quantitative improvements confirm that the high-pressure spool speed provides critical information on the gas-generator state, significantly enhancing the fidelity and robustness of turboprop torque prediction.

The proposed two-stage framework shows that (i) BO-based architecture search outperforms EGD in designing FNN models for turboprop engines, and (ii) optimized ANN models that take into account both flight-condition and core-speed variables can achieve very accurate torque (and therefore thrust) estimates across a wide range of operating conditions. In the future, efforts will focus on refining and extending the proposed framework for high-fidelity flight simulation environments. Accurate real-time predictions of turboprop torque and thrust are critical for pilot training systems and advanced aircraft performance emulation modules.

References

[1] J. D. Mattingly, *Elements of gas turbine propulsion*. Reston, VA, USA: American Institute of Aeronautics and Astronautics (AIAA), 2005.

[2] P. G. Hill and C. R. Peterson, *Mechanics and thermodynamics of propulsion*, 2nd ed. Reading, MA, USA: Addison-Wesley, 1992.

[3] G. F. C. Rogers, H. Cohen, and P. J. Sherwin, *Gas turbine theory*, 6th ed. Harlow, U.K: Pearson Education, 2017.

[4] F. Yildirim Dalkiran and M. Toraman, "Predicting thrust of aircraft using artificial neural networks," *Aircraft Engineering and Aerospace Technology*, vol. 93, no. 1, pp. 35-41, 2021. https://doi.org/10.1108/AEAT-05-2020-0089

[5] M. Zaag, R. M. Botez, and T. Wong, "CESSNA citation X engine model identification using neural networks and extended great deluge algorithms," *Incas Bulletin*, vol. 11, no. 2, pp. 195-207, 2019. https://doi.org/10.13111/2066-8201.2019.11.2.16

[6] M. Sabzehali, A. H. Rabieeb, M. Alibeigia, and A. Mosavi, "Prediction of the energy and exergy performance of F135 PW100 turbofan engine via deep learning," *arXiv preprint arXiv:2208.12028*, 2022. https://doi.org/10.1016/j.enconman.2022.115775

[7] S. Vladov, R. Yakovliev, O. Hubachov, J. Rud, and Y. Stushchanskyi, "Neural network modeling of helicopters turboshaft engines at flight modes using an approach based on “black box” models," in *Proceedings of the Information Technology and Implementation (IT&I-2023) (pp. 116–135). Kyiv, Ukraine: CEUR Workshop Proceedings*, 2023.

[8] R. P. Andrianantara, G. Ghazi, and R. M. Botez, "Performance model identification of the general electric CF34-8C5B1 turbofan using neural networks," *Journal of Aerospace Information Systems*, vol. 20, no. 12, pp. 831-848, 2023. https://doi.org/10.2514/1.I011220

[9] Y. S. Chati and H. Balakrishnan, "Statistical modeling of aircraft engine fuel flow rate," in *30th congress of the international council of the aeronautical science*, 2016, pp. 1-10.

[10] K. Kayaalp and S. Metlek, "Prediction of burning performance and emissions indexes of a turboprop motor with artificial neural network," *Aircraft Engineering and Aerospace Technology*, vol. 93, no. 3, pp. 394-409, 2021.

[11] O. O. Dursun, S. Toraman, and H. Aygun, "Modeling of performance and thermodynamic metrics of a conceptual turboprop engine by comparing different machine learning approaches," *International Journal of Energy Research*, vol. 46, no. 15, pp. 21084-21103, 2022.

[12] M. K. Psaropoulos, M. M. Kaimasidou, K. I. Papadopoulos, V. G. Gkoutzamanis, P. Giannakakis, and A. I. Kalfas, "Surrogate modelling of a Turboprop engine performance," in *Turbo Expo (Vol. 88766, p. V001T01A039). American Society of Mechanical Engineers*, 2025.

[13] T. Baklacioglu, O. Turan, and H. Aydin, "Dynamic modeling of exergy efficiency of turboprop engine components using hybrid genetic algorithm-artificial neural networks," *Energy*, vol. 86, pp. 709-721, 2015. https://doi.org/10.1016/j.energy.2015.04.025

[14] Airbus Defence and Space, "C295 military aircraft," 2025. https://www.airbus.com/en/products-services/defence/military-aircraft/c295

[15] S. S. Haykin, *Neural networks and learning machines*, 3rd ed. Upper Saddle River, NJ, USA: Pearson Education, 2009.

[16] C. M. Bishop, "Pattern recognition and machine learning." New York, USA: Springer, 2006.

[17] G. Dueck, "New optimization heuristics: The great deluge algorithm and the record-to-record travel," *Journal of Computational physics*, vol. 104, no. 1, pp. 86-92, 1993. https://doi.org/10.1006/jcph.1993.1010

[18] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical programming*, vol. 45, no. 1, pp. 503-528, 1989.

[19] H. Gao, D. Wen, and Y. Liu, "Hybrid training of neural networks with L-BFGS and adaptive gradient methods for better generalization," *Neurocomputing*, vol. 453, pp. 34-47, 2021.

[20] B. McCollum, P. McMullan, A. J. Parkes, E. K. Burke, and S. Abdullah, "An extended great deluge approach to the examination timetabling problem," *Proceedings of the 4th multidisciplinary international scheduling: Theory and applications 2009 (MISTA 2009)*, 2009.

[21] E. S. Sin and N. S. M. Kham, "Hyper heuristic based on great deluge and its variants for exam timetabling problem," *arXiv preprint arXiv:1202.1891*, 2012.

[22] A. B. Mosbah, R. M. Botez, and T.-M. Dao, "New methodology combining neural network and extended great deluge algorithms for the ATR-42 wing aerodynamics analysis," *The Aeronautical Journal*, vol. 120, no. 1229, pp. 1049-1080, 2016. https://doi.org/10.1017/aer.2016.46

[23] A. B. Mosbah, R. M. Botez, S. Medini, and T.-M. Dao, "Artificial neural networks-extended great deluge model to predict actuators displacements for a morphing wing tip system," *INCAS Bulletin*, vol. 12, no. 4, pp. 13-24, 2020. https://doi.org/10.13111/2066-8201.2020.12.4.2

[24] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Cambridge, MA, USA: MIT Press, 2006.

[25] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," presented at the Advances in Neural Information Processing Systems 25 (NIPS 2012) (pp. 2951–2959). Lake Tahoe, NV, United States, 2012.