ARQII Publication
http://arqiipubl.com/ams
APPLICATIONS OF
MODELLING AND SIMULATION
eISSN 2600-8084
VOL 9, 2025, 324-337

Enhancing Time Series Prediction with Hybrid AFSA-TCN: A Unified Approach to Temporal Data and Optimization

Nur Alia Shahira Mohd Zaidi, Zuriani Mustaffa* and Muhammad 'Arif Mohamad

Faculty of Computing, Universiti Malaysia Pahang Al-Sultan Abdullah, 26600 Pekan, Pahang, Malaysia

*Corresponding author: zuriani@umpsa.edu.my

Submitted 10 January 2025; Revised 13 July 2025; Accepted 01 September 2025; Available online 12 September 2025.

Abstract: Time series data, with its sequential dependencies presents a unique challenge for traditional machine learning methods such as Random Forest (RF), Support Vector Machines (SVM), and Decision Trees (DT), which often struggle to capture temporal patterns effectively. In contrast, Temporal Convolutional Networks (TCNs) have proven to be highly accurate in addressing these challenges. This study focuses on forecasting the Remaining Useful Life (RUL) of batteries, a critical task for applications such as electric vehicles, renewable energy, and battery management. The dataset used in this study is a battery RUL dataset retrieved from an open-source platform Kaggle, which consists of more than 15,000 rows of time series data. The study introduces a hybrid model that integrates TCN with Artificial Fish Swarm Algorithm (AFSA), a bio-inspired optimization technique designed to fine-tune TCN parameters. The results show that AFSA-TCN achieves outstanding performance with an R-squared (R^2) score of 0.9992, Mean Absolute Error (MAE) of 4.8200, Root Mean Squared Error (RMSE) of 9.0840 and Mean Absolute Percentage Error (MAPE) of 5.48%. Further comparisons with other AFSA-based hybrid models, including AFSA-SVM, AFSA-Gated Recurrent Unit (GRU), and AFSA-Artificial Neural Network (ANN), reveal that AFSA-TCN offers superior prediction accuracy, efficient tuning, and adaptability to the battery RUL dataset. The results highlight the robustness and effectiveness of the AFSA-TCN hybrid model in addressing complex challenges in time series forecasting and optimization, particularly for RUL estimation.

Keywords: Artificial fish swarm algorithm; Optimization; Remaining useful life; Temporal convolutional network; Time series.

1. INTRODUCTION

The transition towards clean mobility solutions is a critical key in addressing global challenges related to climate change and sustainable development. Transportation that relies on fossil fuels has contributed significantly to the formation of smog, a harmful chemical pollutant in the atmosphere [1]. The negative effects of fossil fuels, which are non-renewable energy sources, are increasing, prompting greater focus on developing transportation systems that minimize environmental impact while enhancing efficiency and accessibility. The implementation of regulations and incentives to encourage citizens to adopt cleaner transportation options has begun to prompt governments around the world to move toward smart cities [2]. Clean mobility encompasses various technologies, including electric vehicles (EVs), renewable energy, and other innovations that promote the use of clean energy. A study in India highlights that transitioning to EVs can significantly reduce emissions, particularly when these vehicles are powered by sustainable energy sources [3]. An economic opportunity has emerged with innovations in battery technology and charging infrastructure to make EVs more accessible and affordable. According to a report by the International Energy Agency (IEA) in 2024, EV sales have surpassed 14 million units, with 70% accounted for by battery EVs in the 2023 stock [4]. This rise indicates that EV sales surged by 14% globally, highlighting the increasing adoption of eco-friendly transport alternatives and contributing to key global sustainability targets, including the UN's Sustainable Development Goals (SDGs). Nowadays, the demand for EV batteries has grown exponentially and is expected to increase by 17-fold by 2030 [5]. Additionally, the connection between clean energy and EV batteries is essential for realizing the potential of sustainable transportation.

In the context of EVs and other battery-powered vehicles or transportation, battery RUL is the most crucial metric for estimating the duration or number of cycles a battery can perform effectively before any replacement is required. Effectively understanding and predicting the RUL of a battery are vital for maximizing the performance, reliability, and cost-effectiveness of EVs. This understanding allows for proactive maintenance and better resource management to improve vehicle lifespan and

enhance the impact of advertising on consumer satisfaction toward EVs [6]. There are several factors affecting battery RUL, such as usage patterns like frequent deep discharge [7], temperature [8], and more. Recent studies show that higher temperatures affect battery cells, and a greater depth of discharge leads to greater degradation of battery cells [9]. Monitoring parameters like charge level, heating, voltage, and health enables users to prevent unexpected breakdowns, improve safety, and optimize energy consumption which indirectly ensures better battery performance and longevity.

As people increasingly rely on battery-powered systems, the importance of accurate battery RUL prediction has become a rapidly evolving field with various implications. There are a few challenges in battery RUL prediction, such as real-time adaptation, where the developed model must adapt to drastically changing conditions and real-time usage patterns. Obtaining high-quality data across various operating conditions can also be challenging when developing a suitable predictive model. Time series plays an important role in predicting battery RUL, as it changes over the battery's operational life, and industries have started to adopt smarter energy management systems. It has been proven over the past decade that time series analysis is a powerful tool that revolutionizes decision-making processes in various sectors [10].

Several methodologies are employed to estimate and predict RUL, each with unique advantages and limitations. Empirical models use historical data, and statistical techniques are simple to implement, offering predictive insight with less complexity but may not capture the complex interactions affecting the battery and may lack the robustness of models [11][12]. Physics-based models simulate the physical and chemical processes that occur within the battery. These provide detailed insight but require significant computational resources and detailed knowledge of battery chemistry. Data-driven models have gained popularity since machine learning is able to analyze large datasets to identify patterns and predict RUL, while offering accurate predictions that can adapt to new data. However, they require substantial data for the training process. Combining elements of empirical, physics-based, and data-driven methods forms hybrid models, which aim to capitalize on the strengths of each method to provide a balanced solution for RUL prediction by providing both accuracy and adaptability.

An innumerable dataset, like time series data, is suitable and fit to apply machine learning as a prediction model [13]. The performance of machine learning improves on a specific task over time by learning from data fed, which enables the algorithms to recognize and learn from patterns within that data [14]. However, there are a few traditional machine learning techniques like Random Forestt (RF), Support Vector Machine (SVM), and Decision Tree (DT) that are not suitable for handling sequential dependencies, such as time series data. Hence, TCN was introduced [15], which can handle sequential data like time series. Integrating robust machine learning models like TCN with optimization techniques further enhances their ability to handle complex datasets, including time series data. This synergy ensures not only improved prediction accuracy but also efficient parameter tuning for better performance.

This combination of machine learning and optimization forms a powerful framework for addressing the limitations of traditional approaches, enabling models like TCN to not only process sequential data effectively but also achieve optimal performance through precise parameter tuning. Optimization is a key component for modern problem solving, which is widely used to find the best solutions to challenges by maximizing or minimizing goals while meeting certain requirements. As technology advances, optimization plays an important role in machine learning, supply chain, and financial planning. Optimization also involves forecasting which helps to search for the best parameters of machine learning models to achieve accurate prediction. In this study, optimization is used to enhance the model's ability to set the best parameters for machine learning techniques.

A significant limitation in existing studies is the lack of an integrated framework that effectively combines temporal feature extraction with adaptive hyperparameter optimization. Models such as Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and TCN heavily rely on well-tuned hyperparameters that are manually tuned, which are computationally expensive and challenging to find the optimal parameter combination. Although models like TCN are potentially able to capture long-term dependencies through dilated convolution, they still depend on fixed architecture choices which can lead to underfitting or overfitting without adaptive tuning. Predicting RUL is crucial for applications like EVs, renewable energy storage, and battery management systems. The aim of this study is to deliver a precise and efficient prediction of battery RUL when using a hybrid model that combines the strength of Artificial Fish Swarm Algorithm (AFSA) to optimize the parameters of TCN [16]. The AFSA, inspired by the foraging behavior of fish, is a bio-inspired optimization algorithm known for its strong global search capability, adaptability, and efficiency in handling complex optimization problems. By integrating optimization techniques, the hybrid model can improve performance, solution quality, and adaptability to various datasets.

The key contributions of this study are the development of a hybrid time series prediction model that combines the strengths of AFSA and TCN in predicting battery RUL. The proposed framework introduces several enhancements to predictive modeling. The key contribution of this paper is as follows:

i. The integration of AFSA provides an automated mechanism to explore and fine-tune the parameters of TCN that enable more robust and efficient model performance with lower computational cost. This creates a novel hybrid architecture for time series prediction.
ii. AFSA-TCN operates as fully integrated, adaptive system where AFSA throughout the training, it continuously refines the TCN architecture in response to real-time performance feedback. This dynamic adjustment initiates a unified self-adaptive training process.
iii. The hybrid AFSA-TCN approach enhances the detection of both short-term and long-term trends, resulting in lower error rates and higher reliability in critical applications like battery RUL prediction.

Compared to existing work, the application of TCN offers powerful architecture for sequence modeling and time series

prediction, unlike classical models such as RF, SVM, and DT that require handcrafted features and struggle to capture long-range temporal dependencies. Furthermore, integrating an external algorithm like AFSA provides a more adaptive and globally optimized solution rather than using built-in optimizers like Adam and Stochastic Gradient Descent (SGD). These traditional optimizers often get trapped in local minima and are sensitive to parameter settings which AFSA offers potentially better generalization to fine-tune parameters of machine learning. Additionally, this study compares the hybrid model with other machine learning methods to evaluate and identify the most effective approach for predicting battery RUL.

The remainder of this paper is organized as follows. Section 2 reviews the related work and provides the theoretical background. Section 3 describes the details of the proposed hybrid model, the AFSA-TCN algorithm and datasets used in this study. Section 4 presents the results, comparing the AFSA-TCN with other methods and discussing its effectiveness, accuracy, and performance in predicting battery RUL. Conclusions are presented in Section 5.

2. RELATED WORK

2.1 Related Work

Time series forecasting has been a fundamental research area due to its wide-ranging applications, including financial prediction [17], weather forecasting [18], energy demand estimation, and industrial process monitoring. Over the years, various methodologies have been developed, evolving from traditional statistical models to advanced deep learning techniques. The early methods in time series prediction mostly relied on statistical models such as in the study in [19] using Autoregressive Integrated Moving Average (ARIMA) to predict next-day electricity prices, the Exponential Smoothing (ETS) experiment in [20] to forecast mixed-frequency multivariate time series, and the study in [21] performing Seasonal Decomposition of Time Series (STL) using Loess of the cells' proportion for flood detection. These models, which presume a linear relationship are practical for stationary and moderately complex datasets. ARIMA models are widely used in financial prediction because their stable assumptions allow them to successfully capture trends and cyclic behavior. However, these models generally struggle with nonlinear or high-dimensional time series data [22][23]. This limitation in the traditional statistical models for time series analysis has driven the exploration of machine learning and deep learning models.

Machine learning models were introduced to handle the non-linear and multidimensional time series since they are equipped with advancements in computational capabilities. Common techniques include Support Vector Regression (SVR) [24], RF [25], and Gradient Boosting Machines (GBM). These models use feature engineering to detect temporal connections indirectly such as in a study [26] that implements three machine learning models which are Logistic Regression (LR), RF, and XGBoost to predict the fire risk. In another study conducted in [27], LSTM, Gated Recurrent Unit (GRU), and RNN were adopted to forecast wind power based on time series data. Previously, in the health sector, a study in [28] achieved high accuracy in heart disease prediction using the RF technique and excellent accuracy when implementing a simple supervised machine learning algorithm. This demonstrated that traditional machine learning was effective in time series analysis through feature engineering and supervised learning. However, recent advancements in machine learning have introduced specialized architectures such as TCN which are designed for sequential data processing like time series.

An effective architecture has emerged for sequential data processing and time series prediction where TCN was introduced, which became an alternative to Recurrent Neural Networks (RNNs) and their variants like LSTM. TCN was broadly applied in time series forecasting and anomaly detection. Recently, a study in [29] showed an accurate prediction when adopting TCN techniques for human drowsiness detection based on ECG analysis compared to a drowsiness detection approach using Convolutional Neural Network (CNN) and LSTM. Similarly, the TCN algorithm gave an encouraging result in agriculture, which is rice crop yield prediction with limited running time, as studied in [30], compared to machine learning and deep learning models, showing TCN can outperform them. While TCN and other machine learning techniques like SVM and LSTM have revolutionized pattern recognition and prediction tasks across various domains, their practical implementation is often constrained by the complexity of model configuration. This includes the need for meticulous hyperparameter tuning which is an important phase in determining model performance and a computationally demanding process.

The effectiveness of machine learning techniques like TCN, SVM, and LSTM depends heavily on the process of hyperparameters optimization which is often complex and time-consuming [30][31]. Their architecture requires careful human oversight in setting up crucial parameters that can significantly impact their learning capability and performance. For example, TCN involves unique architecture features that extend beyond basic parameter adjustment, such as the use of dilated convolutions, causal padding, and residual connections which require careful design and configuration [32]. On the other hand, SVM requires setting parameters like the kernel type and margin to help map data into higher dimensions where patterns become easier to separate [33]. Traditionally, the parameters were adjusted through trial and error, which takes computing time and power. These challenges necessitate the use of optimization techniques that are able to automate the parameter tuning and maximize model performance.

Optimization was initially developed to solve problems in fields like economics, engineering, and operations research, where the importance of finding an efficient solution to complex problems is well recognized. In the context of machine learning, optimization focuses on adjusting model parameters to minimize errors and improve predictive accuracy since model performance depends on these parameters. There are a few techniques in optimization, such as Gradient Descent (GD) introduced by the study in [34], with its variants like Stochastic Gradient Descent (SGD) and Mini-Batch Gradient Descent. The inspiration from natural processes like Genetic Algorithms [35], Particle Swarm Optimization (PSO) [36], Ant Colony Optimization [37], and Differential Evolution (DE) [38] has been used in machine learning as evolutionary algorithms that

simulate processes like evolution or swarm behavior to explore the solution space. As a result, various optimization algorithms have been proposed to support machine learning models in achieving better performance through efficient parameter tuning which has led to the development of hybrid model approaches that apply optimization techniques tailored for machine learning tasks.

Recently, many studies have explored a hybrid approach that combines optimization strategies with machine learning. For instance, a study conducted in 2023 [39] showed that the accuracy of SVM gradually increases when combining an optimizer using PSO and a modified PSO algorithm. Another hybrid model, PSO-SVM, was also studied in [39] to predict the RUL of aircraft engines. Meanwhile, a study conducted in [40] demonstrated a few optimization techniques with Extreme Learning Machine (ELM) to predict goat milk purity. The study compares the optimization used, which are Genetic Algorithm (GA), PSO, AFSA, and Improved Artificial Fish Swarm Algorithm (IAFSA), showing that the hybrid models AFSA-ELM and IAFSA-ELM achieved the best results. These two optimizations present a quick and accurate way to identify the optimal combination for the ELM parameters. The effectiveness of using AFSA as an optimization algorithm in the hybrid model is also presented in [41], [42], and [43].

In this study, a hybrid model combining optimization and machine learning approaches was used for predicting time series data, which is the battery RUL dataset. The optimization method used in this study is AFSA, which focuses on adjusting model parameters to minimize errors and improve predictive accuracy. The forecasting model highlighted in this study is TCN due to its ability to forecast time series data compared to traditional recurrent models like LSTM and GRU. Other benchmarking machine learning techniques also combined with AFSA were conducted to compare with TCN. Hence, the AFSA-TCN model was developed to predict a time series dataset which is the battery RUL.

2.2 Temporal Convolutional Network

As a significant advancement in sequence modeling, the TCN was introduced by a researcher [16] and later refined [32], which was originally built on principles from CNNs to handle sequential data. TCN can provide an effective alternative to RNNs and their variants. In RNNs, they process sequences step-by-step and rely on retaining information from previous steps which is prone to losing information over long sequences.

The architecture of TCN is derived from the standard convolutional network principles: causal convolutions and dilated convolutions. Causal convolutions are incorporated in TCN to ensure that the network can only utilize information from present and past inputs, preserving the causal structure of time-dependent data and preventing data leakage from the future. Dilated convolutions, where gaps between kernel elements effectively expand their receptive field exponentially with depth without increasing the number of layers. This architecture allows TCN to adequately capture long-term dependencies within fewer layers.

The structure of TCN consists of several components which are causal convolutions and dilated convolutions that work together, allowing TCN to handle sequential data efficiently. Figure 1 illustrates the TCN architecture built for time-series forecasting which consists of an encoder-decoder. In the encoder process, input sequences are analyzed using layered convolutional layers with increasing dilation factors, allowing the network to efficiently capture temporal dependencies across various time scales. In the model, dilated convolutions allow the receptive field to grow exponentially while maintaining computational efficiency. The last encoder output, h_t, represents a condensed summary of the sequence until time t. The decoder employs this to generate predictions for future time steps. The algorithm incorporates *ResNet* to refine h_t and a dense layer to convert it into expected outputs, including the next step value ($\hat{y}_{t+1}$) and an optional multi-step forecast ($\hat{y}_{t+\Omega}$).

The TCN structure ensures causality, with prediction at any time step depending only on current and historical data. This model successfully simulates long-term dependencies in time-series data using dilated convolutions, residual connections, and parallel processing. These architectural features make TCN particularly suitable for applications such as forecasting RUL, State of Change (SOC), and State of Health (SOH).

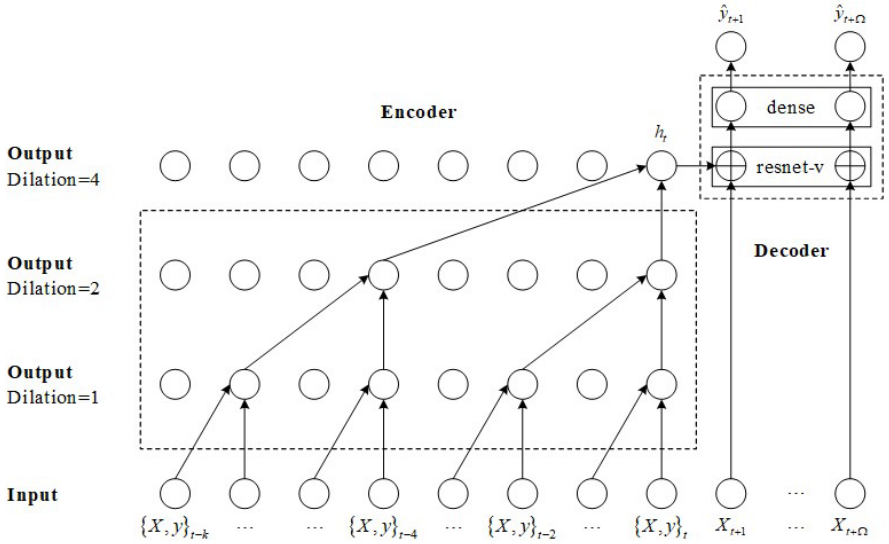

Figure 1. TCN structure diagram [16].

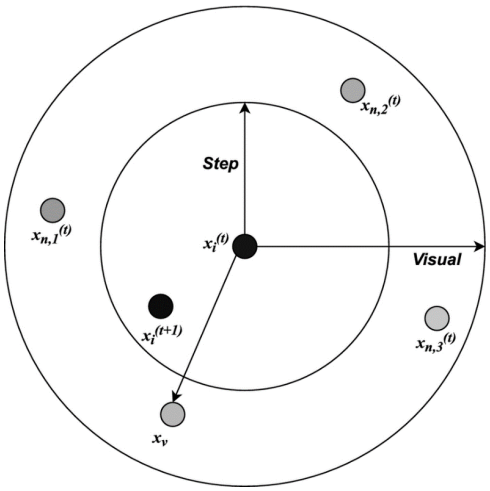

Figure 2. AFSA structure diagram [44].

2.3 Artificial Fish Swarm Algorithm

The AFSA is a population-based optimization algorithm that was inspired by swarming, clustering, and foraging behaviors that fish naturally exhibit. The objective of the algorithm is to identify an optimal solution by simulating natural interactions, and each fish in the search space represents a possible solution. Three primary behaviors, namely prey, swarming, and following, were incorporated into the algorithm. In prey behavior, fish move randomly within their visual range toward areas with higher food concentration, which correlates to better fitness values in the objective function. Fish that engage in swarming behavior use the collective intelligence of the swarm to move toward the center of the group if the average fitness of the group is better than their current position. Meanwhile, fish can efficiently take advantage of potential solutions in following behavior by allowing fish to approach a neighbor with the highest fitness within their visual range. Fish move randomly to keep the search space diverse if there is no better option. These behaviors are controlled by parameters such as visual range, step size, and crowding factors to balance exploration and exploitation [45].

The AFSA refers to the candidate solution as an Artificial Fish (AF). The structure of AF and its surroundings is illustrated as shown in Figure 2. Each AF denoted by $\vec{x} = (x_1, x_2, \dots, x_d)$, exists in a d-dimensional continuous space. The Step (s) parameter limits the maximum Euclidean distance an AF can travel within a single iteration. Meanwhile, the Visual (v) parameter determines the maximum vision range (Euclidean distance). A position inside this range is referred to as $\vec{x}_v$. In addition, $\vec{x}_{n,i}$, where i as a neighbor of the n-th AF within the range of view of the i-th AF, means that the Euclidean distance between them is less than v. Each AF searches within hypersphere centered at its position, with a radius equal to v. AF move one step toward it when they identify a better position that has a better fitness value within the hypersphere.

The AFSA algorithm begins by randomly initializing the population of n Artificial Fish within the search space. These AFs explore their environment using three primary behaviors which are preying, swarming, and following. The bulletin board is used by the algorithm to store the best solution discovered so far. The AF with the highest fitness value is compared to the recorded best solution when the iteration reaches the end. The bulletin board is updated accordingly when the new position outperforms the current best fitness. The iteration continues until the predefined termination condition is satisfied.

3. METHODOLOGY

This study aims to highlight the ability of integrating AFSA with TCN. The AFSA algorithm is selected for the main optimization due to its advantages in handling complex problems, making it highly effective in hybrid model approaches [44]. The key strength of AFSA is its fast convergence which enables it to find an optimal solution quickly. Additionally, AFSA is insensitive to initial parameter settings, as it does not require precise tuning to achieve the best performance and it has strong error tolerance to escape local optima and find a better solution [46]. Recent studies have shown that AFSA-based optimization combining self-organizing neural networks has delivered superior accuracy in cloud resource management compared to an approach using the PSO algorithm [47]. AFSA is also widely used for feature selection and parameter tuning which presents superior performance [48][49]. Hence, this study experiments on AFSA-TCN as a compelling choice over other AFSA-based hybrid models for time-series applications and machine learning integration.

3.1 Data Collection

The dataset used in this study was the battery RUL dataset, which captures the performance and degradation characteristics of the battery over its charge and discharge cycles to predict the RUL. Table 1 shows the sample of the battery RUL dataset retrieved from the Kaggle website [50]. This dataset consists of Cycle Index, Discharge Time (s), Decrement 3.6-3.4 V (s), Maximum Voltage Discharge (V), Minimum Voltage Charge (V), Time at 4.15 V (s), Time Constant Current (s), Charging Time (s) and Remaining Useful Life (RUL). The Cycle Index represents the specific cycle operation, with each row detailing data for one cycle. The Discharge Time (s) estimate the duration of the battery sustaining a discharge, which decreases with

aging and indicates capacity loss. The Decrement 3.6-3.4 V reflects the time for the voltage to drop between 3.6-3.4 V, highlighting the rate of degradation as the battery ages. Meanwhile, the Maximum Voltage Discharge and Minimum Voltage Charge indicate the maximum voltage during discharge and minimum voltage during charging. Both will degrade due to internal resistance and chemical wear. The time at 4.15 V is the duration of the battery maintains a voltage of 4.15 V during charging, while Time Constant Current captures the constant current phase's duration, both of which shift as the battery deteriorates. The Charging Time represents the estimated time required to recharge. It will constantly increase as the battery ages and becomes less efficient. Lastly, RUL quantifies the predicted remaining cycle before it reaches the end of life. This dataset is important for training and testing a hybrid model for this study which is AFSA combined with TCN. The AFSA optimization will be used to optimize the parameters used in TCN to predict the RUL value.

Table 1. Sample of battery RUL dataset.

Cycle_Index	Discharge Time (s)	Decrement 3.6-3.4 V (s)	Max. Voltage Discharge, (V)	Min. Voltage Charge, (V)	Time at 4.15 V (s)	Time Constant Current (s)	Charging Time (s)	RUL
11	435251.5	263086.1	4.267	3.086	269.984	443700	443700	1102
12	3228.58	1135.349	3.689	3.485	5033.076	5969.89	5969.89	1101
13	6019.9	1058.28	4.045	3.475	5053.843	5980.77	5980.77	1100
14	6026.59	1049.488	4.047	3.477	5046.43	5966.82	5966.82	1099
15	6008.07	1065.372	4.045	3.48	5033.076	5954.47	5954.47	1098
16	423271.4	168773.3	4.27	3.108	219924	430028.8	430028.8	1097
17	2261.34	883.2	4.038	3.901	1949.664	2922.69	6070.11	1096
18	2259.46	883.199	4.042	3.373	5181.377	6161.38	9310.98	1095
19	2256.61	878.4	4.042	3.374	5181.375	6154.37	9296.64	1094
20	2252.83	873.601	4.043	3.374	5174.334	6147.33	9243.58	1093

3.2 Data Pre-processing

Before training and testing the AFSA-TCN model, the RUL datasets must undergo pre-processing to ensure optimal performance during learning. In this study, the raw dataset is scaled by applying Min-Max normalization to transform the numerical features into a consistent range of (0, 1). Normalization is important in predictive modelling to prevent any single feature with a larger value from dominating the learning process, which leads the model to overemphasize its importance regardless of their actual predictive power. Without normalization, the optimization process becomes inefficient and unstable, resulting in longer training times. The Min-Max normalization formula is expressed in Equation (1):

$$Z_i = \frac{X_i - min(x)}{max(x) - min(x)} \tag{1}$$

where X_i is the i-th original value in the dataset, $min(x)$ is the minimum value of the feature in the dataset, and $max(x)$ is the maximum value of the feature in the dataset. The result, Z_i is the normalized value of each feature that is scaled to fall within the 0 to 1 range. The sample of normalized data from Table 1 is presented in Table 2.

Table 2. Min-max normalization of battery RUL dataset.

Cycle_Index	Discharge Time (s)	Decrement 3.6-3.4V (s)	Max. Voltage Discharge, (V)	Min. Voltage Charge, (V)	Time at 4.15V (s)	Time Constant Current (s)	Charging Time (s)	RUL
0.0088	0.4542	0.8215	0.9273	0.0472	0.0016	0.5038	0.5038	0.9726
0.0097	0.0034	0.4958	0.4894	0.3412	0.0210	0.0068	0.0068	0.9718
0.0106	0.0063	0.4957	0.7591	0.3339	0.0211	0.0068	0.0068	0.9709
0.0115	0.0063	0.4957	0.7606	0.3353	0.0210	0.0068	0.0068	0.9700
0.0124	0.0063	0.4957	0.7591	0.3375	0.0210	0.0068	0.0068	0.9691
0.0132	0.4417	0.7042	0.9295	0.0634	0.8973	0.4883	0.4883	0.9682
0.0141	0.0024	0.4955	0.7538	0.6478	0.0084	0.0033	0.0069	0.9673
0.0150	0.0023	0.4955	0.7568	0.2587	0.0216	0.0070	0.0106	0.9665
0.0159	0.0023	0.4955	0.7568	0.2594	0.0216	0.0070	0.0106	0.9656
0.0168	0.0023	0.4955	0.7576	0.2594	0.0216	0.0070	0.0105	0.9647

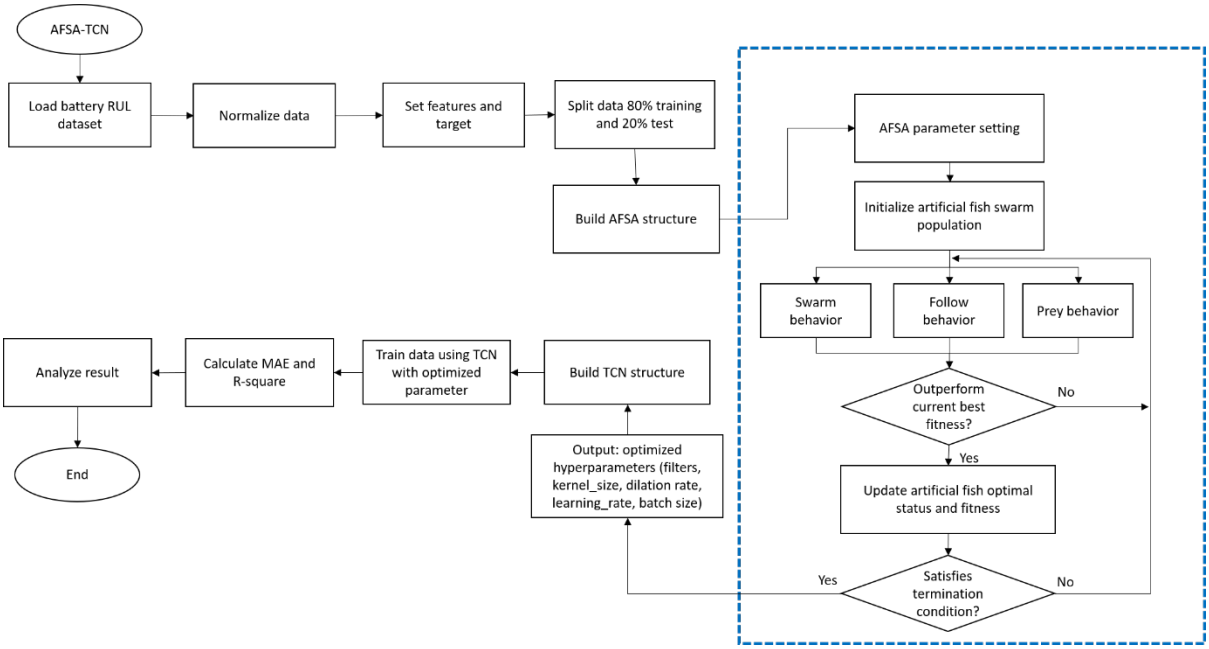

Figure 3. Flowchart of AFSA-TCN.

3.3 Training and testing

The battery RUL dataset shall be divided into a training and testing subset after the features and target are set to evaluate the model's generalization capability. A split ratio of 80% from the dataset used for training, where the model learns the underlying pattern and relationship of the dataset, and 20% is used for testing purposes to assess the model's performance. This division helps the model learn from a substantial portion of the dataset while reserving a separate subset for unbiased performance assessment.

3.4 Hybrid Model – AFSA-TCN

Figure 3 shows the flowchart of the AFSA-TCN model. The flowchart starts with the battery RUL dataset being loaded and read, which serves as the input data. This dataset contains important information about battery RUL, with a sample shown in Table 1. Once the dataset is loaded, the data will undergo normalization using MinMaxScaler to ensure features are scaled uniformly from 0 to 1. This step is important to eliminate bias and prevent any data from being dominated in the learning process due to scale differences. After normalization, the features and target are defined, where the is target is RUL and the rest are the features. Next, the dataset is split into training and testing sets, with 80% and 20%, respectively. The AFSA structure is built, which performs the optimization process to search for the best hyperparameters. The AFSA operates by initializing a population of artificial fish, each representing a candidate solution. The algorithm iteratively evaluates the fitness of each solution, then updates the position of artificial fish based on their movement (prey, swarm, following) and produces an optimal set of hyperparameter output. The output of AFSA, which is hyperparameters including filters, kernel_size, dilation_rate, learning_rate, and batch_size. After optimal parameters were determined, the TCN structure was built using these parameters. Then, TCN is trained using the training dataset. The performance of the TCN model is evaluated using MAE and R-Square (R^2). The trained AFSA-TCN produces output, which is accuracy and errors in prediction. In this study, the R^2 and MAE metric are used to evaluate and analyze the result in the last step. The effectiveness of the AFSA-TCN framework for battery RUL prediction is compared with other machine learning algorithms. The process concludes with optimized parameters and model performance which can be used for further analysis. This comprehensive approach ensures that the model is well-tuned and capable of making accurate predictions, highlighting the effectiveness of the hybrid model AFSA and TCN.

3.5 Implementation

The implementation of the hybrid model AFSA-TCN was developed in Python, focusing on predicting the battery RUL using TCN after being optimized by AFSA. The related libraries for machine learning and deep learning tasks were imported. The libraries used are TensorFlow and Keras for neural network implementation, scikit-learn for data preprocessing and evaluation metrics, Pandas for handling the dataset, and lastly, Numpy for numerical operations. Firstly, after all libraries were imported, the dataset which is the battery RUL was loaded and preprocessed using MinMaxScaler to scale all data in the range between 0 and 1, ensuring that the model can be analyzed efficiently during training.

Once the data was scaled, the features (X) and target (y) are separated. The features are reshaped into a 3-dimensional format (samples, timesteps, features) to comply with the input format required by the TCN model. Next, the data was split into training and testing sets with a ratio of 80:20, respectively. The TCN model is defined as the *build_tcn_model* function using the Sequential API from *Keras*. The model consists of a TCN layer as the primary component, which is configured with hyperparameters such as the number of filters, kernel size, learning rate, and dilation rate. The TCN layer also uses causal

convolutions as padding to preserve the order of the input data and dilated convolutions to expand the receptive field. In the Dense layer that is also included in TCN, the model includes two Dense layers with 64 and 32 neurons. The final Dense layer, which the third output the single prediction value. This model was then compiled using the Adam optimizer with custom learning and MAE as the loss function.

Before training the TCN model with hyperparameters, the model employs AFSA techniques to search for the best parameters. The AFSA class will iteratively evaluate the fitness of each fish using the objective function. Within this function, a TCN model is built with the candidate hyperparameters, trained on the training dataset, and evaluated on the validation set using MAE. The fish with the best performance is selected and the population is updated with their position based on random movement within a defined search space. This process sets a maximum of 20 iterations until the best hyperparameters are found. After finishing the AFSA iterations, the best hyperparameters were used to train the TCN model using the training data. *EarlyStopping* is also used to monitor the validation loss and prevent overfitting by halting the process when the performance is not improving. Lastly, the final trained model is evaluated using the test dataset to compute the test loss and test MAE. The predictions are generated by the model, and the result is compared to the actual target values. The prediction was evaluated using the R^2 score, MAE, RMSE, and MAPE which are calculated to measure the accuracy and errors.

3.6 Performance Evaluation Metric

Metrics like MAE, RMSE, and R^2 are widely used to evaluate the accuracy and quality of predictive models. In this study, MAE, RMSE, and R^2 were used to evaluate the battery RUL prediction. All metrics serve different purposes and provide valuable insights into the performance of a model.

3.6.1 Mean Absolute Error (MAE)

MAE measures the average magnitude of the absolute differences between the predicted values and the actual values. The formula for MAE is shown in Equation (2):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2}$$

where n is the number of observations, y_i represents the RUL actual value of the i-th observation, and $\hat{y}_i$ is the RUL predicted value of the i-th observation.

3.6.2 Root Mean Squared Error (RMSE)

RMSE [51] is a performance metric that calculates the square root of the mean squared deviation between predicted and observed values. Due to the b squaring of differences, RMSE penalizes larger errors than MAE which making it more responsive to outliers. The formula for RMSE is shown in Equation (3):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3}$$

3.6.3 Mean Absolute Percentage Error (MAPE)

MAPE is a common and popular metric used to evaluate the accuracy of predictive models, especially in time series prediction. MAPE calculates the average percentage error between the predicted RUL and the actual RUL value. The formula for MAPE is expressed in Equation (4):

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{4}$$

3.6.4 R-squared (R^2)

On the other hand, R^2 is known as the coefficient of determination, which quantifies the extent to which variance in the target variable is predictable from the independent variables. The range is from 0 to 1, where a value closer to 1 indicates that the model explains most of the variance in the data. The formula for R^2 as in Equation (5):

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2} \tag{5}$$

where $\bar{y}_i$ is the mean of all actual value. The sum which known $\sum_i (y_i - \hat{y}_i)^2$ is the numerator of Residual Sum of Squares (RSS) and $\sum_i (y_i - \bar{y}_i)^2$ is the denominator of Total Sum of Squares (TSS).

Before evaluating the prediction models using MAE, RMSE, MAPE, and R^2, it is important to ensure the data is denormalized to the original scale if normalization was applied during pre-processing. In this study, the dataset was normalized using Min-Max normalization, which scales the data to a range 0 to 1, and it was transformed back to its original value to ensure meaningful and comparable results.

3.7 Benchmarking Techniques

There are a few benchmarking machine learning techniques to compare with AFSA-TCN, which include SVM, GRU, and ANN. The SVM is a supervised machine learning algorithm that is mostly used to perform classification and regression problems. The core of SVM is to find the ideal hyperplane that best separates data points from distinct classes in a high-dimensional environment. In classification problems, this hyperplane serves as the decision border between classes, where it maximizes the nearest data points called support vectors. In non-linear data, SVM uses kernel functions to transform non-linear data into a higher-dimensional space that allows for linear separation. SVM is robust to overfitting, particularly in high-dimensional datasets, but it may require careful tuning of parameters like kernel type, regularization term, and kernel-specific parameters. In time series prediction, SVM is mostly used as a regression model to predict the future value of time series within a specified margin of error while minimizing the deviation for data points outside this margin.

The GRU is a type of RNN architecture. It was designed to address the limitations of standard RNNs, which are the difficulty in learning long-term dependencies in sequential data. GRU is also designed to handle problems like the vanishing gradient during backpropagation, like LSTMs. In contrast, GRU is considered simpler in design and computationally more efficient than LSTM. This is achieved by using two key gates, the update gate, and the reset gate, which are necessary components of the system. The update gate decides how much past information to carry forward to the current state and the reset gate decides how much of the previous state to forget. This capability makes GRU effective in modeling temporal relationships, seasonality, and trends in time series data by learning from historical data.

The ANN is inspired by the structure and functioning of the human brain, which in computational ANN algorithms consists of interconnected layers of neurons that process and transmit information. In an ANN layer, it typically includes an input layer that receives data, one or more hidden layers where the computation happens, and the final output layer that produces the predictions. Each connection between neurons has an associated weight that is adjusted during training to minimize the error between predicted and actual output. The training process involves forward propagation and backpropagation, where the input is passed through the network and errors are propagated backward to update the weights. ANN requires a large amount of data and computational resources, but it is highly versatile and can approximate complex functions. ANN can be effective for time series tasks or when combined with other techniques like feature engineering and data normalization to avoid overfitting during training.

4. RESULT AND DISCUSSIONS

This section presents the performance evaluation of the proposed hybrid AFSA-TCN model for RUL estimation, alongside a comparison with alternative models: AFSA-SVM, AFSA-GRU, and AFSA-ANN. Table 3 outlines the AFSA parameter settings, including population size.

Table 3. AFSA hyperparameter settings.

Hyperparameter	Description	Value/Option
population size, *n*	Number of artificial fish in the swarm represents a candidate solution in problem space.	10
Dimension of the problem, *dim*	The number of variables or dimensions of optimization problem.	5
Maximum iteration	The maximum number of iterations the algorithm runs before termination	20
Visual Distance, *visual*	The maximum distance within which a fish can perceive other fish or food. (Local search radius)	0.2
Step size, *step*	The maximum distance a fish can move in one step.	0.01
Crowd Factor	The maximum acceptable density of fish in a given area. Prevent overcrowding and balance the exploration and exploitation.	1

Table 4. Performance of hybrid model using AFSA.

Model	Mean Absolute Error (MAE)	R-square (R^2)	Mean Absolute Percentage Error (MAPE)	Root Mean Squared Error (RMSE)
AFSA-SVM	10.1242	0.9988	7.96	11.3705
AFSA-GRU	10.8389	0.9973	13.60	16.7051
AFSA-ANN	5.0720	0.9954	5.44	21.8022
AFSA-TCN	**4.8200**	**0.9992**	5.48	**9.0840**

The integration of AFSA to optimize the parameters of the four models further enhances their performance, as shown in Table 4, highlighting the effectiveness of the hybrid machine learning model with optimization techniques. The AFSA-SVM shows a good fit to the data by achieving a high R^2 score of 0.9988. However, its MAE (10.1242) and MAPE (7.96%) were relatively high, and the RMSE (11.3705) was moderate. The AFSA-SVM model is a non-temporal model that lacks an intrinsic mechanism to model time dependencies, which explains its limitation in handling the sequential nature of battery RUL data. Despite optimization using AFSA, the SVM model appears better suited for datasets with less temporal complexity.

Next, the AFSA-GRU model which uses gated recurrent units to capture temporal sequences, achieved an R^2 of 0.9973, indicating a strong alignment with the actual RUL values. However, AFSA-GRU recorded the highest errors with an MAE of 10.8389, RMSE of 16.7051, and MAPE of 13.60%. These values suggest that while GRU captures the general RUL trends, it suffers from inconsistencies in precise prediction due to the GRU's sequential processing nature, which may not match the parallel and hierarchical structure compared to convolutional models

The AFSA-ANN presents mixed results with its R^2 score of 0.9954, the lowest among the four models. Despite that, it performed competitively in error values, especially achieving the lowest MAPE of 5.44%, which was closely followed by TCN. This suggests that AFSA-ANN provides a consistent prediction of RUL. The RMSE for this model was the highest with 21.8022, indicating a higher susceptibility to large prediction errors, which is possibly due to overfitting or a lack of temporal context.

In contrast, the AFSA-TCN shows superior overall performance with an R^2 score of 0.9992, which closely follows the actual RUL distribution, reflecting high predictive performance. This model also achieved the lowest error values with MAE of 4.8200, RMSE of 9.0840, and MAPE of 5.48% which indicate that AFSA-TCN offers a strong model fit and excels in minimizing both average and extreme prediction errors.

The graph shown in Figure 4 depicts the training and validation performance of the AFSA-GRU model. In the loss graph, the training and validation loss decrease significantly during the initial epochs and stabilize later which the model effectively learns from the data and shows good generalization. While in the MAE graph, both training and validation MAE decrease rapidly and remain low and stable, confirming the GRU's ability to perform consistently on both training and unseen data. Figure 5 illustrates the training and validation performance of the AFSA-ANN model. The loss graph steadily declines in training and validation losses, which gradually stabilize over time, showing the model generalized well and does not suffer from overfitting, while the MAE graph shows the lines are almost identical after early epochs, which suggests high consistency and reliable learning.

The graph in Figure 6 highlights the performance of the AFSA-TCN model, which outperforms the other models in training efficiency and generalization without overfitting. The training loss decreases sharply and stabilizes quickly while the validation loss closely follows the training loss, showing minimal signs of overfitting and good generalization. In the MAE graph, both training and validation MAE show a consistent downward trend, with the gap being small, confirming the model maintains high accuracy on unseen data. In comparison, the AFSA-TCN model shows the best performance with minimal overfitting and low loss and MAE values. These results suggest the AFSA-TCN model is well-optimized, accurate, and reliable for battery RUL prediction.

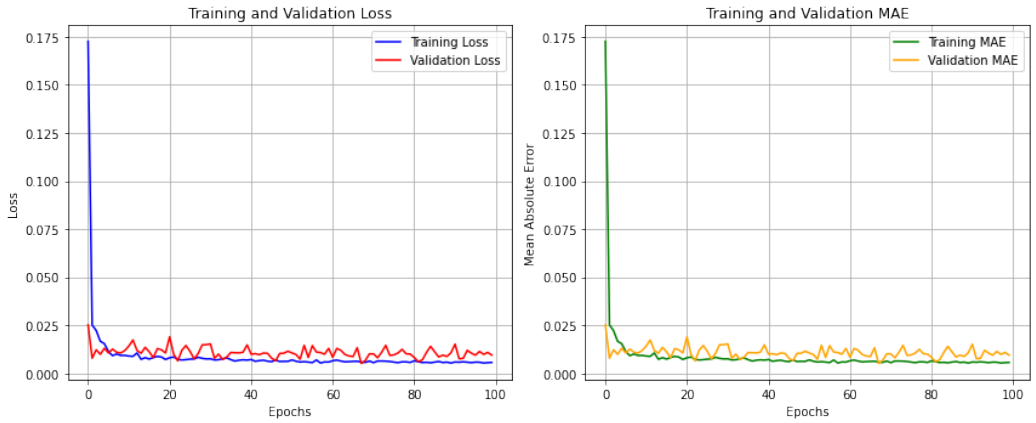

Figure 4. AFSA-GRU training and validation for loss and MAE.

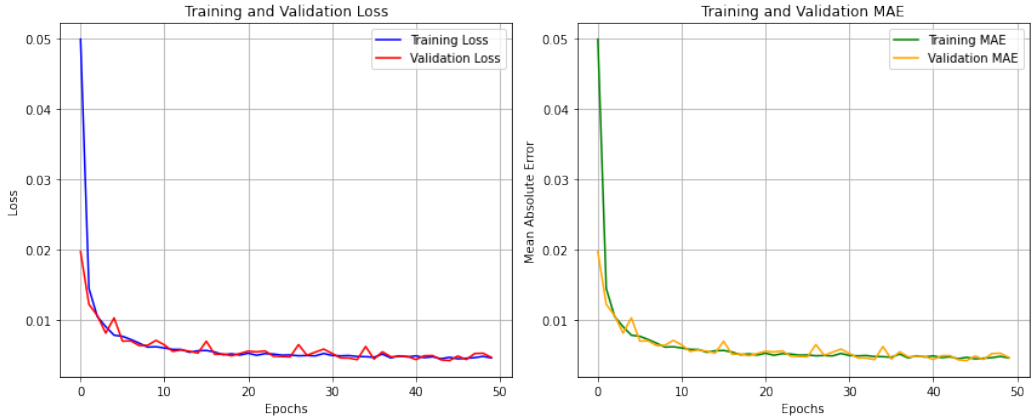

Figure 5. AFSA-ANN training and validation for loss and MAE.

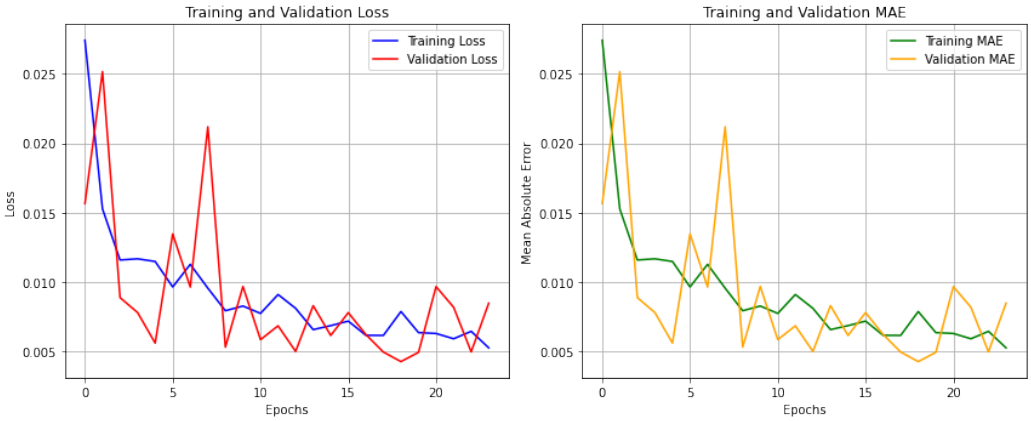

Figure 6. AFSA-TCN training and validation for loss and MAE.

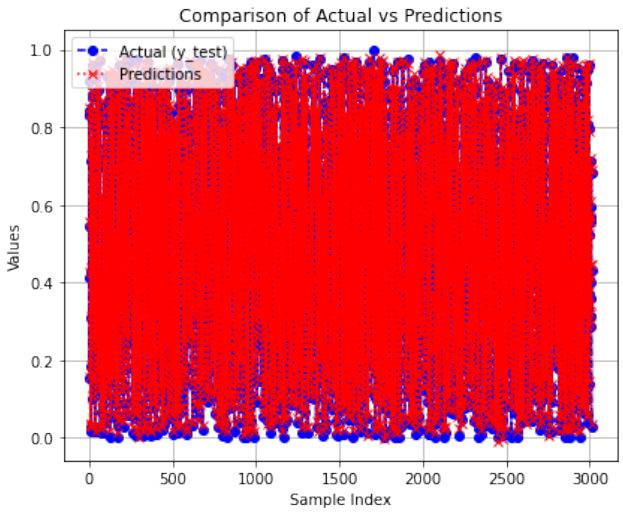

Figure 7. AFSA-SVM comparison of actual and prediction.

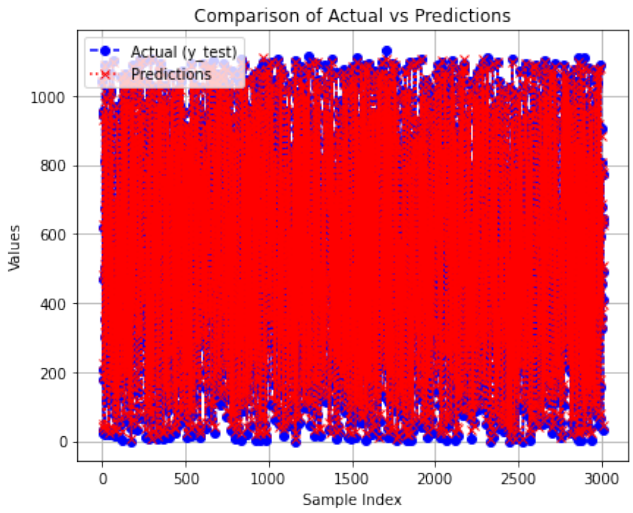

Figure 8. AFSA-GRU comparison of actual and prediction.

Figures 7, 8, 9 and 10 illustrate the comparison of actual values with predicted value in all models. The actual values are presented by the blue dots and predicted values in red. In AFSA-SVM, the close alignment between the actual and predicted values demonstrates that the model can replicate the observed data patterns, where predictions might slightly differ from actual values. In AFSA-GRU, the model shows strong alignment between actual and predicted values with minimal noticeable deviations, which indicates the model performs well in capturing the underlying trends of data and making accurate predictions. The prediction in AFSA-ANN demonstrates better performance, although it might struggle in capturing patterns that lead to less precise prediction compared to AFSA-GRU. Finally, the AFSA-TCN model shows the consistent overlap of actual and predicted values. This demonstrates that the AFSA-TCN hybrid model is the most accurate in predicting values compared to other models, which shows that the model has a strong ability to capture both short-term and long-term dependencies within the data.

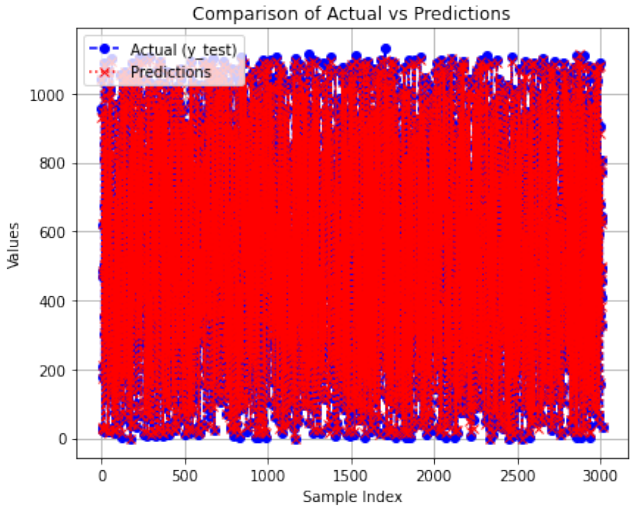

Figure 9. AFSA-ANN comparison of actual and prediction.

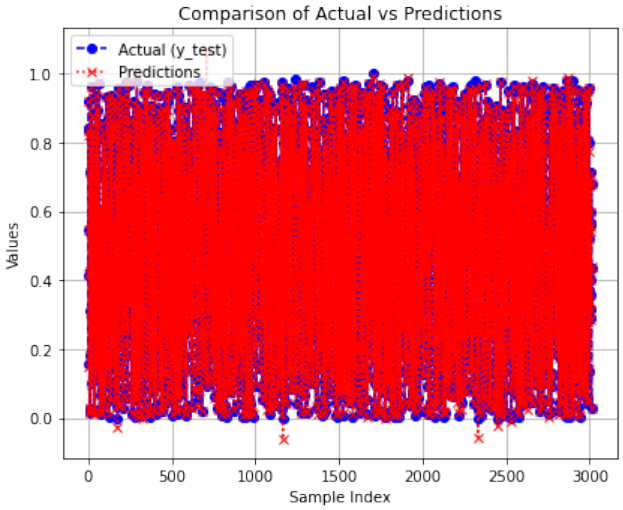

Figure 10. AFSA-TCN comparison of actual and prediction.

AFSA-TCN presents an outstanding performance in prediction since TCN is a specialized architecture for handling temporal data, where it uses dilated causal convolution that enables the model to have a large receptive field while maintaining temporal causality and to learn the temporal patterns over long sequences more efficiently. When applying the battery RUL dataset, the TCN model identifies the subtle signs of degradation throughout the long charge-discharge cycle. Its strength in modelling long-term dependencies makes it highly effective for RUL prediction, where the early detection of capacity loss and aging behavior is crucial.

5. CONCLUSIONS

The result of this study demonstrates that the AFSA-TCN model successfully outperforms other models, proving it to be the most effective approach for time-series prediction using battery RUL datasets. The AFSA-TCN model demonstrates its robustness and precision in handling the temporal and sequential nature of the dataset. The consistent performance in evaluation metrics proves the ability to accurately model complex relationships and trends over time. As the primary focus of this study, the AFSA-TCN achieves the aim of this study by proving its effectiveness as a reliable and accurate predictive tool, surpassing others. This achievement highlights the strength of integrating the AFSA optimization algorithm with TCN techniques for temporal data modelling and positions the AFSA-TCN as a promising solution for predictive tasks.

Despite the strong performance achieved by the AFSA-TCN model, its effectiveness is only validated on the battery RUL dataset, which may limit its generalizability. Therefore, the generalizability of AFSA-TCN to other time-series datasets remains uncertain without empirical validation across diverse domains and conditions. Furthermore, the current implementation of AFSA-TCN relies on a predefined set of hyperparameters and does not include sensitivity analysis conducted. For example, AFSA hyperparameters like population size or step factor can directly impact the efficiency and effectiveness of the optimization process. This may limit the model's ability to reach its full predictive potential across different datasets.

In future work, the application of the AFSA-TCN model can be extended to other domains involving complex time-series datasets, such as healthcare, financial forecasting, and industrial monitoring. This would help to assess the model's adaptability and robustness in diverse environments. In addition, further studies could also explore integrating an alternative hybrid metaheuristic algorithm for optimizing additional hyperparameters to enhance its predictive capabilities. Exploring the integration of attention mechanisms or interpretable layers may also help in making the model more transparent and explainable.

ACKNOWLEDGMENT AND FUNDING

The authors of this paper would like to thank Ministry of Higher Education (MoHE) for funding this work through the research grant FRGS/1/2024/ICT02/UMP/02/2 and Universiti Malaysia Pahang Al-Sultan Abdullah (UMPSA).

DECLARATION OF CONFLICTING INTERESTS

The authors declare no potential conflicts of interest with respect to the research and publication of this article.

REFERENCES

[1] H. Bínová, O. Hykš, M. Hykšová, K. Neubergová, F. Kekula and J. Sadil, Perspective of clean mobility in road freight transport, *Transportation Research Procedia*, 53, 2021, 289-304.

[2] S. Chatterjee, R. Chaudhuri, D. Vrontis, and S. Bresciani, Exploring the effect of government incentives on electric vehicle purchase intention in smart cities, *Journal of Cleaner Production*, 2024, 143841.
[3] I. Sharma and M. K. Chande, Will electric vehicles (EVs) be less polluting than conventional automobiles under Indian city conditions?, *Case Studies on Transport Policy*, 8(4), 2020, 1489-1503.
[4] International Energy Agency, *Global EV Outlook 2024*, International Energy Agency (IEA), Paris, 2024.
[5] K. Liu, Y. Wang, and X. Lai, *Introduction to battery full-lifespan management,* In *Data Science-Based Full-Lifespan Management of Lithium-Ion Battery (Green Energy and Technology)*, Springer, 2022, 1-25.
[6] C. Bi, S. Jin, S. Li and Y. Li, Can green advertising increase consumers' purchase intention of electric vehicles? An experimental study from China, *Journal of Cleaner Production*, 419, 2023, 138260.
[7] E. Karden, Secondary batteries – Lead-acid systems: Automotive batteries, new developments, In *Encyclopedia of Electrochemical Power Sources,* J. Garche, Ed. Amsterdam: Elsevier, 2009, 851-858.
[8] A. Kirchev, Battery management and battery diagnostics, in *Electrochemical Energy Storage for Renewable Sources and Grid Balancing*, P. T. Moseley and J. Garche, Eds. Elsevier eBooks, 2012, 411-435.
[9] P. Lall, V. Soni, G. Sethi and K. Yiang, Effect of high and low storage temperatures, storage duration and varying depth of discharge on coin cell SOH degradation, *2022 21st IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (iTherm)*, San Diego, CA, USA, 2022, 1-8.
[10] A. Nielsen, *Practical Time Series Analysis: Prediction with Statistics and Machine Learning,* 1st ed. United States: O'Reilly Media, 2019.
[11] H. Feng and A. Xu, A data compensation model for predicting SOH and RUL of Lithium–Ion battery, *Journal of Electrical Engineering and Technology*, 19, 2024, 395-406.
[12] L. Hu, W. Wang, and G. Ding, RUL prediction for lithium-ion batteries based on variational mode decomposition and hybrid network model, *Signal Image and Video Processing*, 17, 2023, 3109-3117.
[13] M. Kaur, A. K. Shukla and S. Kaur, An Introduction to Machine Learning in a Nutshell, *2021 10th International Conference on System Modelling & Advancement in Research Trends (SMART),* India, 2021, 17-22.
[14] W. Rand, Theory-interpretable, data-driven agent-based modeling, In *Social-Behavioral Modeling for Complex Systems*, P. K. Davis, A. O'Mahony and J. Pfautz, Eds. Hoboken, New Jersey: John Wiley & Sons, 2019, 337-357.
[15] C. Lea, M. D. Flynn, R. Vidal, A. Reiter and G. D. Hager, Temporal convolutional networks for action segmentation and detection, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017.
[16] N. Zainal, A. M. Zain and S. Sharif, Overview of artificial fish swarm algorithm and its applications in industrial problems, *Applied Mechanics and Materials*, 815, 2015, 253-257.
[17] K. Gajamannage, Y. Park, and D. I. Jayathilake, Real-time forecasting of time series in financial markets using sequentially trained dual-LSTMs, *Expert Systems with Applications*, 223, 2023, 119879.
[18] A. Y. Barrera-Animas, L. O. Oyedele, M. Bilal, T. D. Akinosho, J. M. D. Delgado and L. A. Akanbi, Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting, *Machine Learning with Applications*, 7, 2022.
[19] P. Areekul, T. Senjyu, N. Urasaki and A. Yona, Next day price forecasting in deregulated market by combination of artificial neural network and ARIMA time series models, *2010 5th IEEE Conference on Industrial Electronics and Applications*, Taichung, Taiwan, 2010, 1451-1456.
[20] B. Seong, Smoothing and forecasting mixed-frequency time series with vector exponential smoothing models, *Economic Modelling*, 91, 2020, 463-468.
[21] F. Fichtner, N. Mandery, M. Wieland, S. Growth, S. Martinis and T. Riedlinger, Time-series analysis of sentinel-1/2 data for flood detection using a discrete global grid system and seasonal decomposition, *International Journal of Applied Earth Observation and Geoinformation*, 119, 2023, 103329.
[22] G. E. P. Box, G. M. Jenkins, G. C. Reinsel and G.M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken, New Jersey: John Wiley and Sons, Inc., 2015.
[23] A. A. Alsuwaylimi, Comparison of ARIMA, ANN and hybrid ARIMA-ANN models for time series forecasting, *Information Sciences Letters*, 12(2), 2022, 1003-1016.
[24] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed. New York: Springer, 2013.
[25] L. Breiman, Random forests, *Machine Learning*, 45, 2021, 5-32.
[26] Y. Michael, D. Helman, O. Glickman, D. Gabay, S. Brenner and I. M. Lensky, Forecasting fire risk with machine learning and dynamic information derived from satellite vegetation index time-series, *Science of the Total Environment*, 764, 2021, 142844.
[27] V. Chandran, C. K. Patil, A. M. Manoharan, A. Ghosh, M. G. Sumithra, A. Karthick, R. Rahim and K. Arun, Wind power forecasting based on time series model using deep machine learning algorithms, *Materials Today Proceedings*, 47, 2021, 115-126.
[28] M. M. Ali, B. K. Paul, K. Ahmed, F. M. Bui, J. M. W. Quinn and M. A. Moni, Heart disease prediction using supervised machine learning algorithms: Performance analysis and comparison, *Computers in Biology and Medicine*, 136, 2021, 104672.
[29] A. R. Ismail, D. Sodoyer and F. Elbahhar, Drowsiness detection in humans based on ECG analysis using temporal convolutional network, *2023 International Conference on Automation, Control and Electronics Engineering (CACEE)*, Chongqing, China, 2023, 62-66.
[30] A. Mohan, M. Venkatesan P. Prabhavathy and A. Jayakrishnan, Temporal convolutional network based rice crop yield prediction using multispectral satellite data, *Infrared Physics & Technology*, 135, 2023, 104960.

[31] M. Claesen and B. D. Moor, Hyperparameter search in machine learning, *MIC 2015: The XI Metaheuristic International Conference,* Agadir, Morocco, 2015, 14.
[32] S. Bai, J. Z. Kolter and V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *Machine Learning,* arXiv:1803.01271, 2018.
[33] T. Hofmann, B. Schölkopf and A. J. Smola, Kernel methods in machine learning, *The Annals of Statistics*, 36(3), 2008, 1171-1220.
[34] M. A. Cauchy, Méthode générale pour la résolution des systèmes d'équations simultanées, *Comptes Rendus de l'Académie des Sciences de Paris*, 536-538.
[35] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, 1st ed. The MIT Press, 1992.
[36] J. Kennedy and R. Eberhart, Particle swarm optimization, *Proceedings of the IEEE International Conference on Neural Networks*, Perth, WA, Australia, 1995, 1942-1948.
[37] M. Dorigo and L. M. Gambardella, Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation*, 1(1), 1997, 53-66.
[38] R. Storn and K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, 11(4), 1997, 341-359.
[39] M. P. Behera, A. Sarangi, D. Mishra and S. K. Sarangi, A hybrid machine learning algorithm for heart and liver disease prediction using modified particle swarm optimization with support vector machine, *Procedia Computer Science*, 218, 2023, 818-827.
[40] H. Han, Z. Wang, C. Li, Z. Ma, Z. Yang and X. Ma, Purity detection of goat milk based on electronic tongue and improved artificial fish swarm optimized extreme learning machine, *6th International Federation of Automatic Control (IFAC) Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2019*, 52(30), 2019, 391-396.
[41] A. M. Hilal, A. H. A. Hashim, H. G. Mohamed, M. K. Nour, M. M. Asiri, A. M. Al-Sharafi, M. Othman and A. Motwakel, Malicious URL classification using artificial fish swarm optimization and deep learning, *Computers, Materials & Continua*, 74(1), 2023, 607-621.
[42] W. Jiang, X. Wang and S. Zhang, Integrating multi-modal data into AFSA-LSTM model for real-time oil production prediction, *Energy*, 279, 2023, 127935.
[43] Z. He, L. Liu, and H. Liu, Improved particle swarm optimization algorithms for aerodynamic shape optimization of high-speed train, *Advances in Engineering Software*, 173, 2022, 103242.
[44] F. Pourpanah, R.Wang, C. P. Lim, X. Z. Wang and D. Yazdani, A review of artificial fish swarm algorithms: recent advances and applications, *Artificial Intelligence Review*, 56, 2023, 1867-1903.
[45] X. L. Li, Z. J. Shao and J. X. Qian, An optimizing method based on autonomous animals: fish-swarm algorithm, *Systems Engineering - Theory & Practice*, 22(11), 2002, 32-38.
[46] M. Neshat, A. Adeli, G. Sepidnam, M. Sargolzaei and A, N. Toosi, A review of artificial fish swarm optimization methods and applications, *International Journal on Smart Sensing and Intelligent Systems*, 5(1), 2012, 107-148.
[47] A. Z. Khan, B. M. Rao, J. V. N. Ramesh, E. Muniyandy, E. Bhagyalakshmi, Y. A B. El-Ebiary, and D. N. P. Devadhas, Self-organizing neural networks integrated with artificial fish swarm algorithm for energy-efficient cloud resource management, *International Journal of Advanced Computer Science and Applications (IJACSA)*, 16(2), 2025, 1060-1070.
[48] K. C. Lin, S. Y. Chen and J. C. Hung, Feature selection and parameter optimization of support vector machines based on modified artificial fish swarm algorithms, *Mathematical Problems in Engineering*, 20(1), 2015, 1-9.
[49] D. Jia, Z. Li and C. Zhang, A parametric optimization oriented, AFSA based random forest algorithm: Application to the detection of cervical epithelial cells, *IEEE Access*, 8, 2020, 64891-64905.
[50] A. H. Aboelkhair, RUL analysis & machine learning, https://www.kaggle.com/code/ahmedhatem404/rul-analysis-machine-learnig/notebook, 2024 (accessed 01.08.2024).
[51] I. Hussain, K. B. Ching, C. Uttraphan, K. G. Tay and A. Noor, Evaluating machine learning algorithms for energy consumption prediction in electric vehicles: A comparative study, *Scientific Reports*, 15, 2025, 16124.