The Journal of Supercomputing (2025) 81:646
https://doi.org/10.1007/s11227-025-07105-0

Predicting power consumption of drones using explainable optimized mathematical and machine learning models

Eman I. Abd El-Latif[1] · **Mohamed El-dosuky**[2,3]

Accepted: 20 February 2025

Abstract

Developing an effective energy consumption prediction model has become a central focus in drone-related research. As a result, numerous models are proposed, each differing in complexity and focusing on factors such as thrust and environmental conditions. This paper presents two mathematical models and a machine learning approach to estimate the energy consumption of drones. The Eldosuky model, the first one, examines the effects of weight, thrust, and air-rotor interaction on power consumption. The second model, Eman, takes into account the drone's mass, payload, and external factors like altitude and airspeed. These models are used to simulate drone power consumption in a variety of scenarios, demonstrating their accuracy and utility in real-world situations. Additionally, this paper uses a random forest regressor, a machine learning model that uses actual data to validate energy predictions and simulate drone performance. Then, the accuracy of the model is improved by applying the Fick's law algorithm optimizer, which contains three phases of motion. These phases are referred to as diffusion operator (DO), equilibrium operator (EO), and steady-state operator (SSO). Evaluation metrics are compared both before and after normalization for seven models: Eldosuky, Eman, D'Andrea, Dorling, Stolaroff, Kirchstein, and Tseng. The Eldosuky model has a lower RMSE (500.7558) and MAE (313.3711) before normalization. The Fick's law algorithm optimizer exhibits optimal metrics following normalization, showcasing a noteworthy enhancement with RMSE of 0.24303 and MAE of 0.08805.

Keywords Drones · Fick's law algorithm · Explainable artificial intelligence (XAI) · Mathematical models · Power consumption

1 Introduction

Drones, also referred to as unmanned aerial vehicles, or UAVs, offer several advantages for speedy package delivery compared to trucks. They do not require drivers and, because road systems do not limit them, can frequently travel faster

Extended author information available on the last page of the article

Published online: 23 March 2025

than cars [1]. Humanitarian organizations are actively considering the use of drones in disaster situations [2]. Additionally, drones have a big environmental advantage because they produce fewer emissions than trucks. Despite having many useful features, UAVs' short battery life is a significant drawback [3, 4].

The primary performance metrics for drone delivery are cost, emissions, and range, which are determined by the drone's energy requirements. Precise assessment of drone energy consumption guarantees practical and effective decision-making for drone delivery systems and all other drone applications, such as wireless communications, agriculture, inspections, surveillance, and humanitarian aid efforts. But one of the most important design flaws with UAVs is that their flight times are usually only ten minutes, which is mostly because of their high-power needs and small battery capacity [5, 6]. As a result, accurate estimation of the flight time and range is crucial for dependable operation and energy-efficient route planning.

A reliable energy consumption model is essential for comprehending how different elements—like wind speed, payload, and altitude—affect a drone's energy consumption in various situations. Despite their potential advantages, smaller UAVs usually face severe power constraints that prevent them from completely overcoming the challenges associated with effective operation and independence. In contrast, larger drones—which are usually employed in military applications—benefit from more potent energy sources; however, these advantages come with costs, such as increased weight, decreased agility, and increased noise levels. Energy-efficient operation and efficient mission planning are therefore essential. Nevertheless, it is still difficult to integrate all these factors into a single energy consumption model.

Drone dynamics, environmental conditions (windy, cloudy), drone design (weight, number of rotors, battery weight/efficiency, avionics), and operational requirements (flight time, payload) are just a few of the many variables that affect drone performance. It should also be taken into consideration that drones may need to travel a portion of the way in other vehicles, like trucks or public transportation. Various cargo weights should be considered because they can significantly affect energy consumption models. Wind direction and speed are related to flying speed because the wind's direction can have a favorable or negative impact on the UAV's ability to fly. A UAV may be in a hover, moving vertically, or moving horizontally. It is important to consider the UAV's flying condition and speed. Numerous factors are connected and change during a drone delivery trip.

Numerous research studies have utilized linear, statistical, and mathematical techniques to model energy consumption [7, 16–18]. Regression models have been used in some studies to estimate power consumption based on flight parameters like payload weight, altitude, and velocity [9, 10]. Others have used statistical techniques, such as stochastic modeling and time-series analysis, to forecast energy use in various environmental scenarios. Furthermore, mathematical models based on physics have been created to take propulsion system efficiency, battery discharge characteristics, and aerodynamic drag into consideration [11]. These techniques are generally categorized as "white-box modeling." Most methods take a long time, are highly complex, and are difficult to modify for various

situations [8]. These approaches are less flexible to changing drone configurations and environmental conditions because they rely on preset equations. A machine learning technique used in this study to overcome these constraints uses historical flight data to identify complex patterns of energy consumption [12–14].

Although there are trade-offs with machine learning techniques, especially when it comes to data curation, computational cost, and maintenance, many of these disadvantages can be lessened as the field develops. ML has the potential to overcome its limitations, from increasing algorithm efficiency to implementing more environmentally friendly techniques like federated learning or energy-aware models. In addition to addressing the real-world issues of computational resources and sustainability, machine learning can become an even more potent tool for accurately and efficiently modeling drone energy consumption by continuously improving these techniques [15]. The machine learning approach is required to create an accurate, comprehensive energy prediction model that considers every relevant variable influencing a drone's flight. By using large amounts of data and ensuring higher accuracy and more efficient, less time-consuming algorithms, this approach enables consistent and thorough research results.

This paper aims to improve operational performance, safety, and efficiency by creating an accurate and reliable drone power consumption model. Two mathematical models and a machine learning technique are proposed in this paper to determine the VLOL drone's energy consumption accurately. In the first model, Eldosuky, power consumption is calculated with weight, thrust, and air–rotor interaction. Eman, the second model, considers extra variables like the drone's mass, payload, and environmental factors like airspeed and altitude. Furthermore, the drone's energy prediction is enhanced by the Fick's law algorithm, which ensures a more accurate estimate by considering the three phases of motion. Diffusion operator (DO), equilibrium operator (EO), and steady-state operator (SSO) are the terms for these stages. Seven current models—Eldosuky, Eman, D'Andrea, Dorling, Stolaroff, Kirchstein, and Tseng—are compared to validate our methodology. Before normalization, the Eldosuky model has the lowest MAE (313.3711) and RMSE (500.7558). With an RMSE of 0.24303 and an MAE of 0.08805, our model shows a notable improvement following the application of the Fick's law algorithm optimizer. These outcomes establish a new standard in the field and demonstrate how well our suggested framework works to enhance drone power consumption predictions. The paper's main contributions are:

1. The paper estimates the energy consumption of drones using the Eldosuky and Eman mathematical models.
2. Drone performance is simulated using the random forest regressor, which improves the prediction of energy consumption and performance metrics.
3. As an optimizer, Fick's law algorithm improves the drone's energy prediction by considering the three phases of motion, guaranteeing a more accurate estimate.
4. The study uses explainable artificial intelligence (XAI) methods, like SHAP (Shapley additive explanations), to make sure the random forest regressor's decision-making process is clear and comprehensible.

5. The relationships between various features and energy models are depicted using a correlation heatmap, which helps identify the crucial elements that have a major influence on drone performance and energy consumption.

The remainder of the document is structured as follows: A review of the current drone energy forecasting methodology is provided in Sect. 2. Section 3 provides a thorough explanation of the selected dataset along with an analysis of it. Machine learning and mathematical models are presented in Sect. 4. In Sect. 5, the outcomes and experimental discoveries are displayed. Lastly, Sect. 6 presents the primary conclusion about potential future directions.

2 Drone energy consumption models

This section offers a thorough analysis of current models for UAV power consumption, including important research like the D'Andrea parameters model, Stolaroff's model, and Dorling et al.'s model. Analytical estimates of UAV energy consumption based on propulsion system characteristics, aerodynamic properties, and battery discharge dynamics are the main focus of the D'Andrea parameters model. Stolaroff's model analyzes the environmental effects and energy efficiency of UAVs, especially when compared to traditional modes of transportation. It offers insights into actual power consumption under various operating conditions. To improve the precision of UAV energy predictions, Dorling et al.'s model presents data-driven strategies, such as regression-based and machine learning techniques.

The D'Andrea energy model is developed based on the drone's lift-to-drag percentage [19]. According to the D'Andrea energy model, the drone's mass, airspeed, lift-to-drag ratio, and battery power transfer efficiency are optimized for stable drone flight. There are two versions of the model: one that is standard and does not take wind into account, and another that does take wind into account regarding the headwind that the drone experiences. By including "empty returns," which happen when the drone drops off the payload before making the return flight, the model is further extended. The model makes several assumptions when constructing its energy consumption model. In the absence of wind, D'Andrea offers a definition

$$D = \frac{\sum_{j=1}^{3} r_j V}{370 p \gamma} + p_r \tag{1}$$

where r_j shows the mass of all drone parts, involving the weight of the drone, j is the battery weight, V is the speed of the drone, γ is the effectiveness of power transfer, and p_r is the power needed for drone avionics.

Using the speeds in m/s, the corresponding expression is calculated using Eq. 2, while energy consumption for steady-level is determined using Eq. 3

$$D = \frac{(\sum_{j=1}^{3} r_j) V g}{370 p \gamma} + p_r \tag{2}$$

$$E = \frac{1}{1-\varphi}\left(\frac{(\sum_{j=1}^{3} r_j)Vg}{p\gamma} + \frac{p_r}{V}\right) \tag{3}$$

Figliozzi expands D'Andrea's "steady flight" basic model to take empty returns into account [20]. This model offers a unitless parameter for battery recharged efficiency (γ), models the lift-to-drag ratio based on speed (with (v)), and excludes power for avionics (p_r). When there is no headwind, the total energy consumption rate per unit distance is as follows:

$$E = \frac{1}{2}\left(\frac{g\sum_{j=1}^{3} r_j}{k(v)p\gamma} + \frac{g\sum_{j=1}^{2} r_j}{k(v)p\gamma}\right) \tag{4}$$

The energy consumption rate with a package is the first term in parenthesis, and the energy consumption rate on the return trip without a package is the second term. A model for drone energy consumption based solely on hovering is presented by Dorling et al. [8]. Since there is no airspeed while in hover and the thrust equals the weight force,

$$T_e = g\sum_{j=1}^{3} r_j \tag{5}$$

The energy needed for the drone to hover, according to helicopter theory, is

$$P = \frac{T_e^{3/2}}{\sqrt{2i\partial C}} = \frac{\left(g\sum_{j=1}^{3} r_j\right)^{3/2}}{\sqrt{2i\partial C}} \tag{6}$$

where C is the area of one rotor's spinning blade disk and i is the number of rotors.

$$E = \frac{\left(g\sum_{j=1}^{3} r_j\right)^{3/2}}{v\sqrt{2i\partial C}} \tag{7}$$

In addition to conducting field testing, Stolaroff et al. [11] offer a component model based on the fundamental forces of flight, such as forces of weight, parasites, and generated drags. Stolaroff et al. present a thrust model for forward flight.

$$T = T_e + W = g\sum_{j=1}^{3} r_j + \frac{1}{2}\rho\sum_{j=1}^{3} C_j A_j v^2 \tag{8}$$

Stolaroff et al. employ an energy consumption formula (adapted from Hoffmann et al. [21]) for forward flight or strong wind.

$$D = \frac{T\left(v_i sin \propto + v_j\right)}{\tau} \tag{9}$$

where $\propto$ is the angle of attack, and v_j is the the induced speed and can be calculated as follows

$$v_j = \frac{g\sum_{j=1}^{3} r_j}{2i\rho s\sqrt{\left(v_i cos \propto\right)^2 + \left(v_i sin \propto + v_j\right)^2}} \tag{10}$$

$$\propto = tan^{-1}\left(\frac{\frac{1}{2}\rho\left(\sum_{j=1}^{3} C_j A_j v^2\right)v^2}{g\sum_{j=1}^{3} r_j}\right) \tag{11}$$

$$E = \frac{T\left(v_i sin \propto + v_j\right)}{v_i \delta} \tag{12}$$

A component model originally presented by Langelaan et al. [22] is provided by Kirchstein [23]. It is based on an idealized delivery process that consists of takeoff, the ascent to a cruising altitude, level flight, descent with hovering and landing for delivery. The energy consumption is calculated using the following equation:

$$E = \frac{1}{\mu}\left(\frac{iT}{v} + \frac{1}{2}\rho\left(\sum_{j=1}^{3} C_j A_j v^2\right)\right) + \frac{i_2\left(g\sum_{j=1}^{3} m_j\right)^{1.5}}{v} + i_3\left(\left(g\sum_{j=1}^{3} m_j\right)^{0.5} v\right) + \frac{p_r}{v\mu} \tag{13}$$

3 The proposed model

This paper uses the Eldosuky and Eman mathematical models, to calculate a drone's power requirements, as well as the random forest regressor, to simulate drone performance and validate energy predictions using real data as shown in Fig. 1. The Eldosuky model focuses on basic aerodynamic and thrust-related factors, whereas the Eman model covers a wider range of drone flight conditions, even though it includes additional complexities like payload, altitude, and air drag. Additionally, the Fick's law algorithm optimizer—which uses three phases of motion—is employed. Diffusion operator (DO), equilibrium operator (EO), and steady-state operator (SSO) are the names given to these stages. The evaluation metrics of the following seven models are compared: Eldosuky, Eman, D'Andrea, Dorling, Stolaroff, Kirchstein, and Tseng.

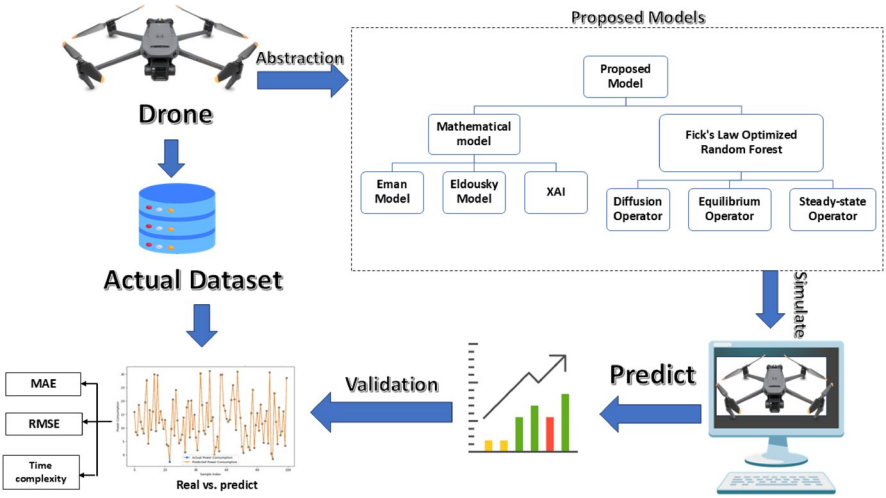

Fig. 1 The proposed model

3.1 Abstraction phase

The process of condensing the real drone system and its dataset into conceptual models is known as abstraction. The drone gathers real datasets from its operations and suggested models capture the main behaviors and phenomena of the system. For simpler simulation and analysis, these models (mathematical and machine learning models) contain only the most important information and leave out any extraneous details.

Information was gathered during 75 flights by electric vertical takeoff and landing (VTOL) unmanned aerial vehicles (UAVs) carrying a variety of payload weights while flying in two distinct configurations and landing. A set of discrete options is used to change the specified parameters between flights: payload of 0, 400 g, and 800 g; altitude of 50 m and 100 m; and cruise speed of 18 m/s and 22 m/s. A wide range of on-board sensors were used to gather data, including an FT Technologies FT205 wind sensor with an accuracy of ± 0.1 m/s and a refresh rate of 10 Hz; a 3DM-GX5-45 GNSS/INS sensor pack with a maximum output rate of 10 Hz and an accuracy of ± 2 m (horizontal), ± 5 m (vertical); and a Mauch Electronics PL-200 current and voltage sensor that can record up to 200 A and 33 V. Table 1 provides a detailed description of the dataset parameters.

The process of running suggested models in a virtual environment to forecast drone behavior is known as simulation, and it begins after abstraction is finished. Based on the abstracted models, the simulation estimates the system's performance under different conditions using mathematical and machine learning models (Fick's law algorithm optimizer).

Table 1 Dataset parameters

Parameters	Description
Flight	An integer that stands for the flight that was completed
Speed	The specified airspeed throughout fixed-wing flight in meters per second
Payload	Mass in grams of the payload placed to the drone
Altitude	Specified height in meters
Battery discharge rate	The battery employed for the test's discharge rate
Temperature	The degree temperature of the air
Humidity	The air's relative humidity, represented as a percentage (%)
Weather station time	The date and time at which the weather station received its weather data
Dew point	The temperature at which dew forms and the air gets saturated

3.2 Mathematical model

Mathematical modeling is essential to comprehending and improving drone flights, despite the complexity brought about by many variables and environmental factors. Models offer an organized method of examining aerodynamics, flight dynamics, and control mechanisms, whereas mathematical functions might find it difficult to account for all real-world uncertainties. They aid in performance optimization, stability improvement under various circumstances, and drone behavior prediction. Furthermore, the development of machine learning algorithms that can adjust to changing environments is facilitated by mathematical models.

The data preprocessing step is an essential part of the machine learning process that builds robust and reliable prediction models. It involves several intricate steps intended to purify unprocessed data and guarantee that it is suitable for modeling. The normalization step is a specific type of feature scaling. The term "normalization" encompasses a range of data scaling strategies, including Max Normalization, Z-Score Standardization, and Min–Max Scaling. These techniques are frequently used to guarantee that each feature makes an equal contribution to the model's functionality. We employ Min–Max Normalization in this study. Implementing Min–Max Normalization is simple and does not involve any intricate computations. The formula is computationally easy because it only scales and shifts the data according to its minimum and maximum values.

Eldosuky and Eman, two mathematical models utilized to estimate a drone's energy requirements, are discussed in Eqs. (14) and (15). These models consider a variety of variables that impact the drone's performance, including thrust, weight, altitude, drag, and airspeed. While the Eman Model includes additional complexities like payload, altitude, and air drag, which makes it more suitable for a wider range of drone flight conditions, the Eldosuky model concentrates on fundamental aerodynamic and thrust-related factors.

3.2.1 Eldosuky model

An important tool for calculating a drone's power needs is the Eldosuky model, which considers important variables like weight, thrust, and air–rotor interaction. It models the nonlinear effect of the drone's weight on power consumption and considers the power required to generate thrust, which is dependent on airspeed and thrust force. The model also accounts for aerodynamic drag, which rises with airspeed, and the drone's propulsion efficiency, which has a big impact on the overall power needed. The Eldosuky model offers a thorough framework for enhancing drone design, forecasting energy usage and enhancing flight performance by combining these components. This model focuses on the ways that thrust, weight, and the air–rotor interaction (via airspeed and rotor area) affect the overall power required.

$$P = \frac{T \times v_a + \beta w^{1.5} + \gamma \rho A v_a^3}{\delta} \tag{14}$$

where P is the total power (w), T is the thrust force (N), w is the weight of the drone, A is the effective area of the drone rotors (m^2), γ is a constant for the aerodynamic, β is constantly related to altitude's effect on air density, v_a is the airspeed (m/s), and δ is the propulsion efficiency of the drone.

3.2.2 Eman model

The Eman model calculates a drone's overall power consumption by considering its mass, payload, and environmental variables like altitude. When calculating power, the model takes into account the drone's and its payload's combined mass, airspeed and drag, and altitude's impact on air density and power consumption. It specifically accounts for propulsion efficiency, altitude effects on air density, and the drag coefficient. This model considers the drone's altitude, drag forces, and payload to give a more complete picture of the power requirements. It considers how the drone's mass and the surrounding environment, including altitude, affect its performance.

$$P = \frac{g\left(m_d + m_p\right) \times v_a + \alpha v_a^3 + \beta\left(h/h_{ref}\right)\left(m_d + m_p\right)}{\delta} \tag{15}$$

where P is the total power (w), g is the gravitational acceleration (9.81 m/s^2), m_d is the payload mass (kg), v_a is the airspeed (m/s), α is the drag coefficient factor, A is the altitude (m), h_{ref} is the reference altitude, β is constantly related to altitude's effect on air density, and δ is the propulsion efficiency of the drone.

The first step in Algorithm 1 involves reading the dataset using a function such as read_csv. Once the data are imported, the MinMaxScaler algorithm is applied to normalize the dataset, scaling the values to a predetermined range, typically between 0 and 1. This ensures that all input features are on a similar scale, improving the performance of the models. After the dataset has been scaled, the Eldosuky model is used to predict energy consumption, initializing key parameters such as air density and gravitational constants. In this model, placeholders for parameters like area and efficiency are represented by (β, γ, and δ). Following this, the Eman model is

applied. These two models are performed to create a thorough mathematical framework that analyzes the normalized data and predicts power output while taking thrust, wind speed, and other relevant factors into account.

Algorithm 1 Mathematical model

Input: dataset from CSV file
Output: calculate power from mathematical models

- read_csv
- Normalize the data using MinMaxScaler algorithm by using the equation:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- Apply Eldosuky Model using the following:
  - Initialize gravitational constant and air density
  - **Set Constants and Parameters**:
  - beta is a placeholder constant, gamma is another placeholder constant, is a Placeholder value representing the area and eta is an Efficiency factor.
  - $P_{Eldosuky} = \frac{T \times v_a + \beta w^{1.5} + \gamma \rho A v_a^3}{\delta}$
- Apply Eman Model using the following equation:
  - $P_{Eman} = \frac{g(m_d + m_p) \times v_a + \alpha v_a^3 + \beta (h/h_{ref})(m_d + m_p)}{\delta}$

3.3 Fick's law optimized random forest

A machine learning algorithm called the random forest regressor creates several decision trees and aggregates their results to generate predictions. The number of trees and additional hyperparameters, like tree depth and minimum sample requirements for splits, are first set. Bootstrap sampling is the process of building each tree during training using a randomly chosen subset of the dataset. A random subset of features is used to split the data at each node, and the process is repeated until a stopping condition is satisfied. When testing, a new input sample is run through each trained decision tree, which generates a regression value, to produce predictions. Using the bagging technique, the outputs from each decision tree are averaged to determine the final prediction. This average prediction is the result of the algorithm. Compared to a single decision tree, this ensemble approach decreases overfitting and increases accuracy [24]. This process of aggregating predictions across multiple trees helps reduce variance and improve the model's ability to generalize to unseen data, which is the core advantage of using the random forest approach over individual decision trees.

The rest of this subsection is dedicated for elaborating on the Fick's law algorithm. Three phases of motion are necessary for Fick's law algorithm to function: diffusion operator (DO), equilibrium operator (EO), and steady-state operator (SSO). The Fick's law algorithm simulates energy transitions in drone motion by utilizing diffusion-based principles, which yields distinct advantages over traditional optimization methods. In contrast to techniques like gradient descent or genetic algorithms,

which concentrate on minimizing error functions, Fick's law considers the three phases of motion to guarantee a more precise estimate of energy consumption.

3.3.1 Diffusion operator

Initially, consider that region u has a greater concentration than the region v. This means that some molecules move from the $u's$ region to the v's, and this will have an impact on the molecules that are still in the u′s region. Given the following formula to calculate the number of molecules moving from the region u to the region v

$$N_{u,v} \approx N_u \times p_1 \times \left(c_1 - c_2\right) + N_u \times c_2 \tag{16}$$

The number of molecules that will move from the u group to the v group is denoted by $N_{u,v}$, and the constants c_1 and c_2 have respective values of 0 and 1.

As molecules $N_{u,v}$ move to a new region, their positions are updated primarily based on the region v's best equilibrium molecule by using:

$$y_{r,u}^{t+1} = y_{ed,v}^{t} + \mathrm{DF}_{r,u}^{t} \times D \times R \times \left(j_{u,v}^{t} \times y_{ed,u}^{t} - y_{r,u}^{t+1}\right) \tag{17}$$

where $y_{ed,u}^{t}$ is the equilibrium position in the region v, the direction factor, denoted by $\mathrm{DF}_{r,u}^{t}$, is either $\{-1,1\}$, which varies arbitrarily and provides a high scanning opportunity within the specified search area as well as an escape from the local optimum, R is a random number between 0, 1, and D is the flow direction that varies over time and is represented by the following equation:

$$D = \exp\left(-c_3\left(F^t - R\right)\right) \tag{18}$$

$$j_{u,v}^{t} = -\mathrm{eff}\frac{dg_{u,v}^{t}}{dy_{u,v}^{t}} \tag{19}$$

where eff means to the effective diffusivity stable which equals 0.1, $\frac{dg_{u,v}^{t}}{dy_{u,v}^{t}}$ denotes the concentration gradient, and $dg_{u,v}^{t}$ and $dy_{u,v}^{t}$ are computed using the following equation

$$dg_{u,v}^{t} = y_{m,v}^{t} - y_{m,u}^{t} \tag{20}$$

$$dy_{u,v}^{t} = \sqrt{\left(y_{ed,v}^{t}\right)^2 - \left(y_{p,u}^{t}\right)^2 + ep} \tag{21}$$

where $y_{m,v}^{t}$ and$y_{m,u}^{t}$ are the mean of molecule position in regions u and v, respectively.

The molecules in u's group (NR) that were still present in the region u adjust their positions in response to navigation through three distinct stages: (1) the equilibrium position within the region u; (2) the equilibrium position within the region u and the problem boundary; or (3) no modification in position. This tactic is simulated using the following equation:

$$y_{r,u}^{t+1} = \begin{cases} y_{ed,u}^{t} & \text{rand} < 0.8 \\ y_{ed,u}^{t} + D \times (R \times (X - L) + L) & \text{rand} < 0.9 \\ y_{r,u}^{t+1} & \text{otherwise} \end{cases} \tag{22}$$

where $y_{ed,u}^{t}$ is the equilibrium position, X, L are the upper and lower boundaries, and R is a random number between [0, 1].

Since the concentration in the region v is higher than in other regions, the molecules in v adjust their positions based on the equilibrium position in the region v, staying in the same area without moving. The boundary of the problem at hand can be determined using the following equation.

$$y_{r,u}^{t+1} = y_{ed,u}^{t} + D \times ((X - L) + L) \tag{23}$$

3.3.2 Equilibrium operator (EO)

This stage is regarded as the transitional stage between exploration and exploitation. The molecular position has been modified by:

$$y_{p,i}^{t+1} = y_{eo,p}^{t} + q_{eo,i}^{t} \times y_{r,u}^{t} + q_{eo,i}^{t} \times \left(m_{eo,p}^{t} \times y_{eo,p}^{t} - y_{i,p}^{t} \right) \tag{24}$$

where $y_{i,p}^{t}$ is the position of the particle in the group i, $y_{eo,p}^{t}$ is the equilibrium location in the group i, and $q_{eo,i}^{t}$ and $m_{eo,p}^{t}$ are computed as follows:

$$m_{eo,p}^{t} = \exp\left(-\frac{F_{i,ed}^{t}}{\left(F_{i,p}^{t} + ed \right)} \right) \tag{25}$$

$$q_{eo,i}^{t} = r_{1}^{t} \times D_{i}^{t} \times \mathrm{DF}_{i,ed}^{t} \tag{26}$$

where $F_{i,p}^{t}$ is the best fitness score in group i, $\mathrm{DF}_{i,ed}^{t}$ is the diffusion rate, and it is computed as follows:

$$\mathrm{DF}_{i,ed}^{t} = \exp\left(-j_{p,ed}^{t} / F^{t} \right) \tag{27}$$

where

$$j_{p,ed}^{t} = -\mathrm{eff} \frac{dg_{i,ed}^{t}}{dy_{p,ed}^{t}} \tag{28}$$

where

$$dg_{i,ed}^{t} = y_{i,ed}^{t} - y_{m,i}^{t} \tag{29}$$

$$dy_{p,ed}^{t} = \sqrt{\left(y_{ed,i}^{t}\right)^{2} - \left(y_{p,g}^{t}\right)^{2} + ep} \tag{30}$$

3.3.3 Steady-state operator (SSO)

The exploitation, or stability phase, is the last stage of the optimization search, during which the molecules update their locations utilizing the following equation:

$$y_{p,i}^{t+1} = y_{ss}^{t} + q_{i}^{t} \times y_{r,i}^{t} + q_{i}^{t} \times \left(m_{i,p}^{t} \times y_{ss}^{t} - y_{i,p}^{t}\right) \tag{31}$$

where y_{ss}^{t} is the steady-state location, $q_{i}^{t}\&m_{i,p}^{t}$ are referred to the relative quantity of the region i, and they can be calculated using the following equation:

$$m_{i,p}^{t} = \exp\left(-\frac{F_{ss}^{t}}{\left(F_{i,p}^{t} + ep\right)}\right) \tag{32}$$

$$q_{i}^{t} = r_{1}^{t} \times D_{i}^{t} \times \mathrm{DF}_{i}^{t} \tag{33}$$

where r_{1}^{t} is a random number between [0, 1], and DF_{i}^{t} is the direction factor

$$\mathrm{DF}_{i}^{t} = \exp\left(-j_{p,ss}^{t}/F^{t}\right) \tag{34}$$

$$j_{p,ss}^{t} = -\mathrm{eff}\frac{dg_{i,ss}^{t}}{dy_{p,ss}^{t}} \tag{35}$$

where

$$dg_{i,ss}^{t} = y_{i,m}^{t} - y_{ss}^{t} \tag{36}$$

$$dy_{p,ss}^{t} = \sqrt{\left(y_{ss}^{t}\right)^{2} - \left(y_{p,g}^{t}\right)^{2} + ep} \tag{37}$$

The Fick's law algorithm (FLA) begins by initializing a set of parameters and randomly generating a population of molecules as shown in Algorithm 2. The population is then divided into two equal groups. For each group, the algorithm calculates the fitness of each molecule, identifies the best molecule in the group, and determines the overall global best solution. The algorithm then enters its main loop, which runs until a predetermined maximum number of fitness evaluations is reached. During each iteration of the loop, different operations are applied based on a parameter called TF. If TF is less than 0.9, the steady-state operator (SSO) is used, where diffusion rates and motion steps are computed to update the molecules' positions. If

TF is less than a random value, the equilibrium operator (EO) is applied, calculating diffusion rates and group relative quantities to update the positions. If neither condition is met, the diffusion operator (DO) is activated, determining the direction of flow and the number of molecules that will move to a new region, and then updating the positions accordingly. Throughout the process, fitness evaluations are updated. When the maximum number of evaluations is reached, the algorithm terminates and returns the best solution found during the optimization process.

Algorithm 2 Fick's law algorithm

Initialization Phase:

- Initializing parameters: C_1, C_2, C_3, C_4, C_5, D.
- Randomly initializing the population X_i(i=1, …, N)

Clustering:

- Dividing the population into two equivalent groups: N_1 and N_2.
- For each group:
  - Computing the fitness for each molecule.
  - Finding the best molecule and the global optimum.

Main Loop (While FES ≤ MAX_{FES}):

- If TF<0.9, use the **Steady State Operator (SSO)**:
  - Calculating diffusion rate and motion step factors (using equations 31 & 32).
  - Updating the individual position (using equation 29).
- Else, if TF<rand, use the **Equilibrium Operator (EO)**:
  - Calculating diffusion rate and group relative quantities (using equations 22 & 21)
  - Updating individual position (using equation 20).
- Else, use the **Diffusion Operator (DO)**:
  - Calculating the direction of flow (using equation 14).
  - Determining the number of molecules moving to the region (using equation 11).
  - Updating individual positions and other molecules (using equations 13, 18, and 19).

Updating Fitness:

- Updating FES and terminating when the maximum is reached.

Return the Best Solution.

4 Mathematical proof

There are four mathematical methods to prove our models. A fundamental relationship for drone motion is established by Newton's second law, which applies to the thrust and weight forces involved in flight. For accurate energy predictions, the rotor–air interaction is modeled by Bernoulli's principle and aerodynamic drag equations. Real-world environmental changes are taken into account

by atmospheric models for altitude effects, which are only utilized in the Eman model. The theoretical models are empirically supported by real-world experiments and drone flight simulations.

4.1 Newton's second law

The forces and power needs acting on the drone are calculated using Newton's second law in both models. The weight and altitude terms take environmental and gravitational factors into account, while the thrust and drag terms derive directly from Newton's law when applied to the drone's motion through the air. According to Newton's second law:

$$F = \mathrm{MA} \tag{38}$$

where F is the net force on the object, M is the mass of the object, and A is the acceleration.

4.1.1 Eldosuky model derivation using Newton's second law

As per Newton's second law, thrust is the force produced by the drone's rotors to offset its weight and drag. The power needed to keep moving forward at airspeed v_a is:

$$P = T \times v_a \tag{39}$$

where T is a function of the force needed to accelerate air downward to produce lift. The weight of the drone W can be calculated as follows:

$$W = \mathrm{MG} \tag{40}$$

The power needed to lift the weight can be computed based on the drone's mass and the thrust needed to stay in the air. Because of the square root dependence of power required for sustained lift, a scaling factor such as $W^{1.5}$ shows that power increases as weight increases. The air density (ρ), effective rotor area (A), and velocity (v_a^3) all affect the amount of power needed to overcome air drag. So, Eq. (41) is used to compute the power required for calculating air drag, which is derived from the drag force, which is calculated using Eq. (42).

$$P_{\text{drag}} = \gamma \rho A v_a^3 \tag{41}$$

$$F_{\text{drag}} = \frac{1}{2} \rho v_a^2 c_d A \tag{42}$$

where c_d is the drag coefficient.

Finally, Eq. 14 combines the power required to overcome aerodynamic drag (drag term), lift the weight (weight term), and maintain forward motion (thrust term).

4.1.2 Eman model derivation using Newton's second law

The total weight of the drone and its payload is equal to $g\left(m_d + m_p\right)$. The drone's mass and speed affect its ability to defy gravity.

$$P_{\text{thrust}} = g\left(m_d + m_p\right)v_a \tag{43}$$

This term, which incorporates the drone's and payload's combined weight, is comparable to the thrust term in the Eldosuky model. Aerodynamic drag is proportional to v_a^3 as in the Eldosuky model, giving rise to the following term:

$$P_{\text{drag}} = \alpha v_a^3 \tag{44}$$

The drone's power needs vary with altitude because of variations in air pressure and density. The term h/h_{ref} scales the amount of power needed according to altitude, where thinner air at higher altitudes means more power is needed.

$$p_{\text{altitude}} = \beta\left(h/h_{\text{ref}}\right)\left(m_d + m_p\right) \tag{45}$$

4.2 Bernoulli's principle and aerodynamic drag equations

The relationship between pressure and velocity in a fluid (like air) is described by Bernoulli's principle. Bernoulli's equation for a steady, incompressible airflow is:

$$P + \frac{1}{2}\rho v^2 + \rho g h = \text{constant} \tag{46}$$

where P is the static pressure, ρ is the air density, v is the airspeed, g is the gravitational acceleration, and h is the altitude. The drag equation provides the drag force that an object traveling through air is going to encounter:

$$F_{\text{drag}} = \frac{1}{2}\rho v_a^2 c_d A \tag{47}$$

4.2.1 Eldosuky model derivation using Bernoulli's principle and drag equation

According to Bernoulli's principle, the pressure difference between the rotor system's top and bottom determines the amount of thrust the rotors produce. The airspeed over the rotors (v_a)is related to the pressure difference (ΔP)

$$\Delta P = \frac{1}{2}\rho v_a^2 \tag{48}$$

The thrust force T needed to maintain the drone's altitude is

$$T = \frac{1}{2}\rho v_a^2 A \tag{49}$$

The following power is needed to produce this thrust:

$$P_{\text{thrust}} = T.v_a = \frac{1}{2}\rho v_a^3 A \tag{50}$$

The weight of the drone is W = mg, and the lifting power required for sustained flight is proportional to the power needed to counteract weight. The empirical model for this term is βW 1.5 βW 1.5. Although this term is not directly affected by Bernoulli's principle, the weight of the drone increases the amount of power required to maintain its mass in the air. The following power is needed to overcome drag:

$$P_{\text{drag}} = v_a F_{\text{drag}} = \frac{1}{2}\rho v_a^3 A C_d \tag{51}$$

The sum of the forces needed for thrust, weight resistance, and drag forces equals the total power needed.

4.2.2 Eman model derivation using Bernoulli's principle and drag equation

The drone and its payload are subject to the following gravitational force:

$$F_g = g\left(m_d + m_p\right) \tag{52}$$

When the drone travels at airspeed v_a, the power needed to overcome this force is:

$$P_{\text{gravity}} = F_g.v_a = g\left(m_d + m_p\right)v_a \tag{53}$$

The power to overcome drag is determined using Eq. (46), just like with the Eldosuky model. The air density reduces with altitude, influencing the amount of power needed to generate lift. According to Bernoulli's principle, to produce the same lift at higher altitudes, the airspeed over the rotors must increase, requiring more power as shown in Eq. (45). Based on the aerodynamic drag equation and Bernoulli's principle, we have deduced the fundamental elements of the Eldosuky and Eman models. The key elements affecting a drone's power consumption while in flight are thrust (pressure variations), drag, weight, and altitude effects, all of which are taken into consideration by the power terms.

4.3 Atmospheric model: the international standard atmosphere (ISA)

Under standard conditions, the air temperature, pressure, and density vary with altitude. This is commonly described by the International Standard Atmosphere (ISA). The principal connections are:

1. In the troposphere, temperature drops linearly with altitude (up to roughly 11 km) as follows:

$$T(h) = T_0 - Lh \tag{54}$$

where $T(h)$ is the temperature at altitude, $T_0 = 288.15k$ is the standard temperature at sea level, and $L = 0.0065k/m$ is the rate at which temperature decreases.

2. Altitude causes an exponential drop in air pressure. The pressure in the troposphere is determined by:

$$P(h) = P_0\left(1 - \frac{Lh}{T_0}\right)^{\frac{Mg}{\mathrm{RL}}} \tag{55}$$

where $P(h)$ is the pressure, P_0 is the standard pressure, g is the acceleration, M is the molar mass, and R is the universal gas constant.

3. According to the ideal gas law, air density additionally reduces with altitude and is influenced by temperature and pressure.

$$P(h) = \frac{P(h)}{T(h)R_{\text{specific}}} \tag{56}$$

where $P(h)$ is the air density and R_{specific} is the gas constant.

The term "altitude effects" in the Eman model represents how power consumption increases with altitude due to decreasing air density. It is a simplified expression of the altitude effects. We observe that air density drastically decreases with altitude using the ISA atmospheric model, resulting in a greater power requirement to maintain the same lift. Higher altitudes require the drone's motors to work harder, which raises the device's total power consumption.

Validation is the process of comparing the model's predictions with the real dataset that the drone collected after the simulation is finished. Measures such as root-mean-square error (RMSE) and mean absolute error (MAE) are employed to ascertain the degree to which the predictions align with actual results. The model is deemed valid if it faithfully captures the behavior of the actual system.

5 Experimental results, discussion, and analysis

5.1 Experimental parameters

This section covers the experimental results used to evaluate the model performance. The following subsections cover the main features of the implementation environment, how the experiments are set up, how the parameters are initialized, and performance metrics. We used Python 3.6.3 for our experiments and the scikit-learn library for the random forest regressor.

5.2 Performance metrics

The details of the results are then presented and discussed. Each experiment is conducted on a PC with a 10th generation Intel® CoreTM i7 processor, a 64-bit Windows 10 operating system, and 8.00 GB of RAM. The root-mean-square error

(RMSE), which computes the average deviation between predicted and actual values, offered a thorough assessment of the model's accuracy. The mean absolute differences between the actual and predicted values were calculated using the mean absolute error (MAE) to estimate the size of the model's prediction errors.

$$\mathrm{RMSE} = \sqrt{\sum_{j=1}^{M} \frac{\left(P_j - A_j\right)^2}{M}} \tag{57}$$

$$\mathrm{MAE} = \frac{1}{M} \sum_{j=1}^{M} \left|P_j - A_j\right| \tag{58}$$

A list of various models and the initial parameters that correspond to them is given in Table 2. The initial parameters for each of the corresponding models are represented by these values, which can be used in theoretical simulations. The starting values for the Eldosuky model are as follows: Beta is set at 0.5, Gamma at 0.3, A at 1, and Eta at 0.9. Alpha is set at 0.1, and Beta is set at 0.05 in the Eman model. Placeholders are used in the D'Andrea model, with mass and velocity initialized at 1 and 10, respectively. The four rotors that make up the Dorling model have a pressure value of 1. With alpha set at 10 and velocity set at 1, the Stolaroff model starts. Lastly, there are three Kappa parameters in the Kirchstein model: Kappa 1 is set at 0.1, Kappa 2 at 0.05, and Kappa 3 at 0.02.

Our models' hyperparameters were selected using both empirical tuning and domain expertise to maximize the algorithms' performance. In order to minimize overfitting and guarantee that the models could identify the underlying patterns in drone power consumption data, the parameters for each model (e.g., Alpha and Beta for Eman; Beta, Gamma, A, and Eta for Eldosuky) were chosen to strike a balance between generalization and model complexity. For example, regularization frequently uses parameters like beta and gamma to help the model adapt to input features. As demonstrated in the D'Andrea model, physical characteristics like mass and velocity were incorporated to more accurately represent the dynamics of drones in the real world, where these factors have a direct bearing on energy usage. Likewise, in the Dorling model, the parameters of pressure and rotor count were essential for simulating drone aerodynamics. To improve the model's learning rate and

Table 2 Initial value for each model

Types of models	Initial parameter
Eldosuky	Beta = 0.5, gamma = 0.3, A = 1, eta, 0.9
Eman	Alpha = 0.1, beta = 0.05
D'Andrea	placeholders' velocity = 10, placeholders mass = 1
Dorling	Pressure = 1, number of rotors = 4
Stolaroff	Alpha = 10, velocity = 1
Kirchstein	Kappa_1 = 0.1, Kappa_2 = 0.05, Kappa_3 = 0.02

regularization terms and avoid underfitting while maintaining peak performance, we also added parameters like Alpha and Kappa coefficients.

The initial parameters for optimizing a model based on Fick's law algorithm are given in Table 3. It has two sets of parameters: additional hyperparameters involving machine learning, such as N estimators, Max depth, and Min samples, and coefficients C1 through C5, which are constants in the Fick's law algorithm equation. Initial values of C1 = 0.5, C2 = 2, C3 = 0.1, C4 = 0.2, and C5 = 2 are assigned to the coefficients. N estimators = 59, Max depth = 20, and Min samples = 1 are the hyperparameters that correspond to the parameters utilized in the optimization procedure. As can be seen in the visualizations comparing actual and predicted power usage, these starting values serve as the evaluation for the optimizer to modify and enhance model's accuracy in predicting power consumption.

5.3 Experimental results

Figure 2 shows histogram of prominent features. It reveals key insights into wind conditions and battery performance. Wind speed exhibits a bimodal distribution, with peaks around 2–5 m/s and 20–25 m/s. Wind angle follows a multimodal distribution, clustering around 100–150°, 250°, and 300°, indicating prevalent wind directions influenced by environmental or seasonal factors. Battery voltage appears normally distributed, centering around 48–49 V, signifying stable performance. However, battery current is right-skewed, with most values between 10 and 15 A but occasional spikes beyond 40A. These insights can aid in optimizing wind energy utilization, ensuring stable power supply, and identifying potential inefficiencies in the system.

Figure 3 shows a thorough analysis of how much power is used by different models as airspeed rises from 0 to 20 m/s. The horizontal axis denotes airspeed, with a range of 0–20 m/s, and the vertical axis represents power consumption, with a maximum of 12,000 watts. A unique color designates each model, as the description demonstrates. The Kirchstein model (yellow line) shows a significant rise in power consumption, which peaks at over 10,000 watts at 20 m/s. This is especially evident above an airspeed of 10 m/s. This implies that the Kirchstein model might be more susceptible to variations in airspeed that necessitate a higher speed-dependent

Table 3 Initial value for Fick's law algorithm optimizer

Parameters	Value
C1	0.5
C2	2
C3	0.1
C4	0.2
C5	2
N estimators	59
Max depth	20
Min samples	1

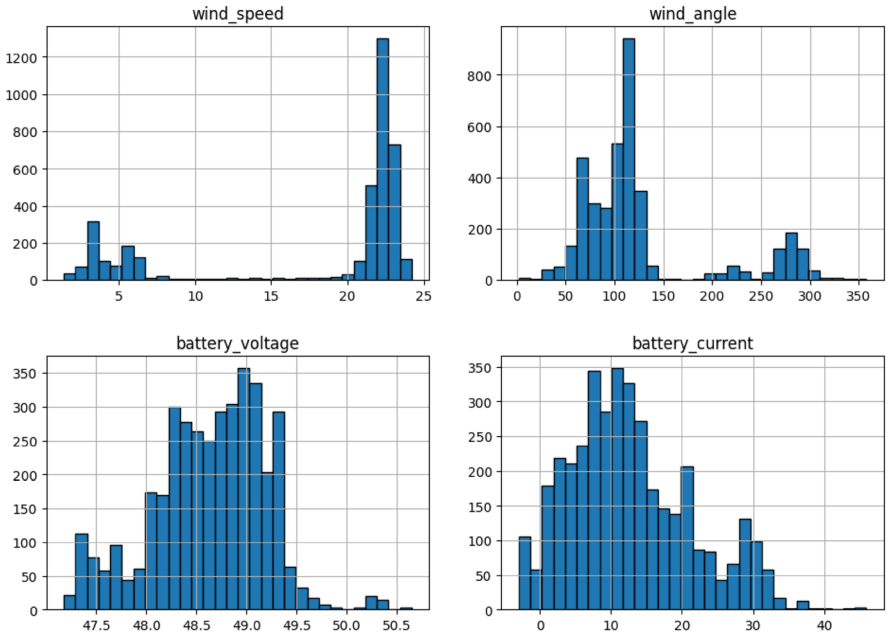

Fig. 2 Histogram of prominent features

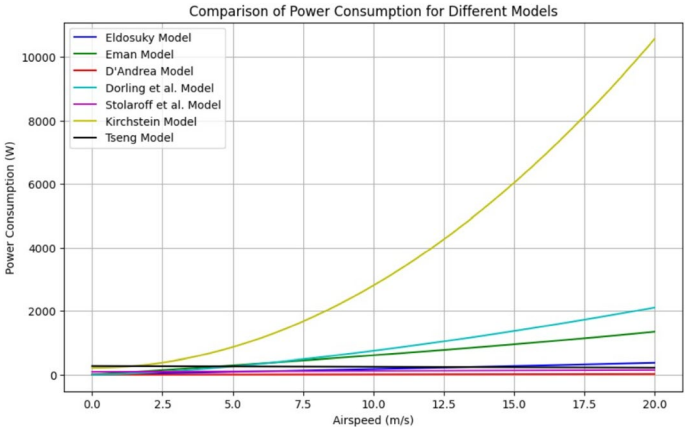

Fig. 3 Power consumption for different models

exponential increase in power. On the other hand, models such as the Tseng (black), Dorling et al. (magenta), D'Andrea (red), Eldosuky (blue), and Eman (green) exhibit significantly reduced power consumption over the whole airspeed range. Their power curves are comparatively flat and stay well below 1000 watts, suggesting different system dynamics assumptions or more effective power usage. The power consumption of the Stolaroff et al. model (cyan line) is moderately higher than that

Table 4 The evaluation metrics before normalization

Types of model	RMSE	MAE
Eldosuky	500.7558	**313.3711**
Eman	1052.887	1011.4343
D'Andrea	658.678	546.0016
Dorling	1835.712	1738.5224
Stolaroff	564.781	432.286
Kirchstein	11,168.926	10,009.0887
Tseng	**491.6604**	374.0259

Tseng's RMSE (491.6604) is the lowest and Eldosuky, on the other hand, has a comparatively lower MAE (313.3711)

Table 5 The evaluation metrics after normalization

Types of model	RMSE	MAE
Eldosuky	0.4317	0.40118
Eman	1.8294	1.62957
D'Andrea	0.3984	0.34816
Dorling	0.5804	0.5545
Stolaroff	3.5908	3.5845
Kirchstein	9.8663	9.85208
Tseng	11.6592	11.65802
Fick's law optimized random forest	0.24303	0.08805

of the models that were discussed earlier, but noticeably lower than the Kirchstein model.

Evaluation metrics for different models are summarized in Table 4 before normalization. Seven models (Eldosuky, Eman, D'Andrea, Dorling, Stolaroff, Kirchstein, and Tseng) are compared in the table based on two standard error metrics: root-mean-square error (RMSE) and mean absolute error (MAE). The RMSE and MAE values are significantly higher before normalization, indicating that the models have difficulty processing the data as is. Kirchstein exhibits the highest MAE (10,009.0887) and RMSE (11,168.926) values. Eldosuky, on the other hand, has a comparatively lower MAE (313.3711) and RMSE (500.7558). Tseng's RMSE (491.6604) is the lowest when compared to Eldosuky, but its MAE (374.0259) is a little higher.

Normalization is essential for ensuring that features with different scales or units contribute equally to the model, which improves convergence speed and overall performance, particularly in models that are sensitive to feature scale, like distance-based models and gradient-based algorithms. We remove any bias that might result from unnormalized features by normalizing the data, ensuring an equitable comparison of the models' performances.

After normalization, two performance metrics—RMSE and MAE—are used to compare various models in Table 5. The Fick's law algorithm optimizer performs best in this model based on both metrics, demonstrating a significant improvement

with RMSE of 0.24303 and MAE of 0.08805. This is a big improvement over the previous version, which had a substantially higher RMSE. Eldosuky, with an RMSE of 0.4317 and MAE of 0.40118, is not far behind D'Andrea, who likewise does well with an RMSE of 0.3984 and MAE of 0.34816. These models indicate good accuracy with low error values. Even with slightly higher errors—an RMSE of 0.5804 and an MAE of 0.5545—Dorling outperforms the other models. However, models like Tseng, Kirchstein, and Stolaroff exhibit noticeably greater error values. Kirchstein and Tseng have even greater RMSE values than Stolaroff, at 9.8663 and 11.6592, respectively. Stolaroff's RMSE is 3.5908. Their MAE values are also high, which suggests that they perform less well than the best models. In conclusion, D'Andrea and Eldosuky are not far behind Fick's law algorithm optimizer as the most accurate model, having the lowest RMSE and MAE values. The remaining models exhibit differing levels of inaccuracy, with Kirchstein and Tseng exhibiting the lowest degree of reliability.

The random forest regressor used in this paper is one of the machine learning models whose interpretability is greatly improved by the SHAP (Shapley additive explanations) values. By assigning a model's output to specific features, SHAP values offer a unified measure of feature importance. They provide insights into the decision-making process by enabling us to comprehend how each feature contributes to a model's predictions. In particular, SHAP values illustrate how different characteristics—like altitude, payload, or environmental conditions—affect the drone's estimated energy consumption. Building confidence in the model's outputs and making the findings useful for practical applications depend on this transparency.

The SHAP values, which show how various features affect a machine learning model's output, are visualized in Fig. 4. The color of each dot indicates the value of the feature, ranging from blue for low to red for high. Each dot represents a

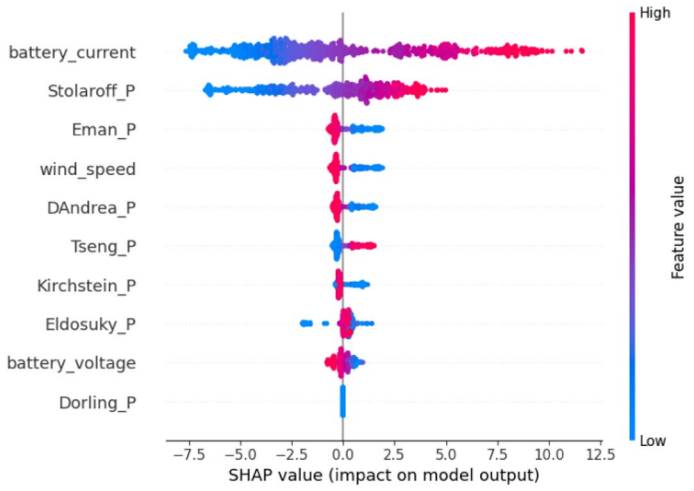

Fig. 4 SHAP value for different models

single observation. The degree to which a feature affected the model's prediction, either favorably or unfavorably, is indicated by the SHAP value on the x-axis.

With a broad range of SHAP values from roughly −7.5 to over 12.5, battery_current is the most influential feature and has a significant impact on the model's output. The model increases the prediction performance when battery_current is higher (red dots), and less when it is lower (blue dots).

Stolaroff_P, Eman_P, and wind_speed are some other features that have significant effects, but they are more concentrated around zero. Kirchstein_P, D'Andrea_P, and Tseng_P show moderate effects with more evenly distributed positive and negative contributions. Eldosuky_P and battery_voltage, on the other hand, exhibit comparatively small effects because of their closely packed SHAP values around zero. Dorling_P appears to have the least impact on the model's predictions; most SHAP values are very close to zero, indicating that it is a minor factor.

Overall, the plot shows how the model's output is influenced by each feature, with battery_current having the greatest influence and Dorling_P having the least. The color variation and dot spread show how various feature values affect the model predictions, either positively or negatively.

An essential tool for comprehending the connections between various energy models and the characteristics that affect drone energy consumption is a correlation heatmap. A heatmap helps identify important factors that should be prioritized for optimization by displaying the correlation between variables and highlighting which features are most strongly associated with drone performance and power requirements.

The correlation coefficient values are represented by the color intensity and hue on the map, which range from −1.00 (ideal negative correlation) to 1.00 (ideal positive correlation). Lighter hues denote weaker correlations; red shows strong positive correlations, and blue demonstrates negative correlations. The heatmap shows correlations between different energy models (e.g., Eldosuky_P, Eman_P, D'Andrea_P) and energy-related variables such as wind_speed, battery_voltage, and battery_current, as presented in Fig. 5. For instance, battery_current exhibits a more varied correlation pattern, with some positive and some negative correlations, whereas wind_speed has a strong positive correlation (near 1) with many models. Based on comparable behavior, the energy models appear to cluster, which could indicate that certain models are impacted by comparable energy system factors.

Figure 6 uses Fick's law optimized random forest to compare "Actual Power Consumption" and "Predicted Power Consumption." The graph shows different data points over a series of measurements or time steps with "Power Consumption" on the y-axis and a "Sample Index" on the x-axis. The actual power consumption is shown in the figure by the blue circle, and the predicted power consumption is shown by the orange line.

With only slight variations between actual and predicted values, the two lines closely follow one another, suggesting that the optimized model performs well in forecasting power consumption trends. High power consumption variability across the sample index is demonstrated by the fluctuations in both lines, and the alignment indicates that the model correctly depicts these dynamics for the majority of the samples.

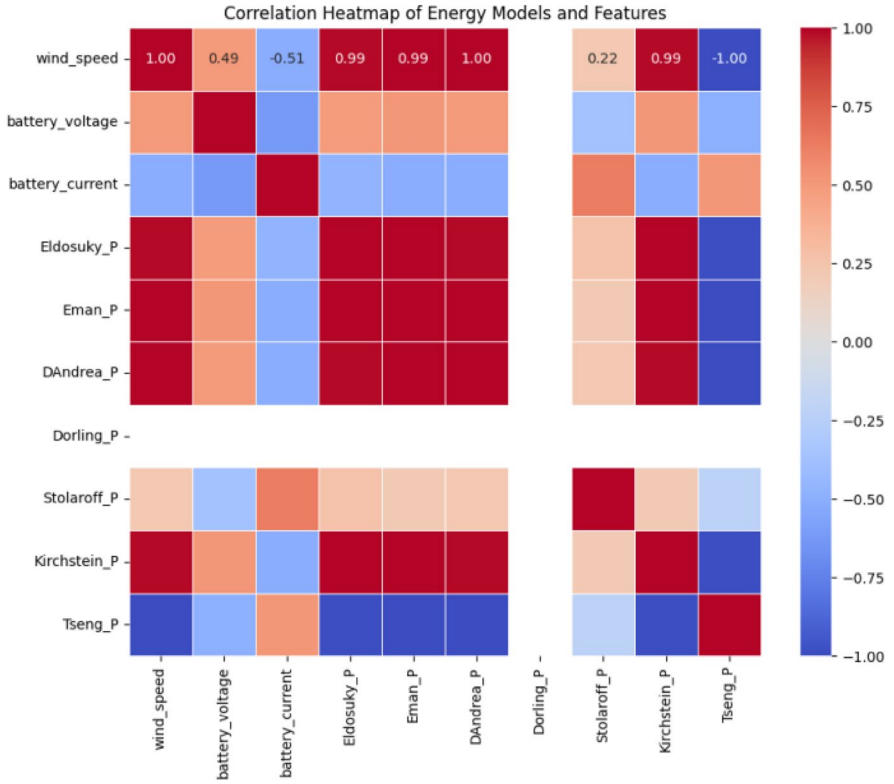

Fig. 5 Correlation heatmap of energy models

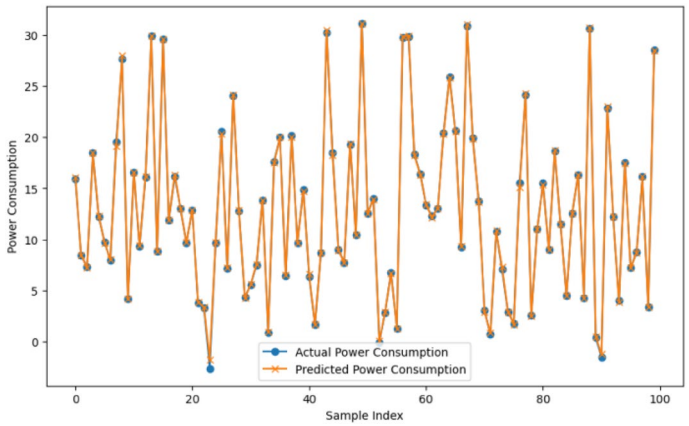

Fig. 6 Fick's law optimized random forest vs actual power consumption

5.4 Time analysis

As shown in Table 6, the overall time complexity of Fick's law algorithm is dominated by the main optimization loop, where each iteration involves updating the population and calculating fitness, both of which take O(N) time. Since the loop runs until a maximum number of function evaluations (MAXFES) is reached, the total complexity of the main loop is O(MAXFES·N). The initialization phase includes setting parameters and initializing the population, and the clustering phase, where the population is divided, both take O(N). After each iteration, the fitness is updated, in O(N). Therefore, the algorithm's overall time complexity is O(MAXFES·N), reflecting its linear growth with both the population size and the number of function evaluations.

Table 7 shows the execution times of two random forest model variants: a Fick's law optimized random forest and a standard random forest without optimization. While the optimized version takes 48.59 s, it is approximately 12 times slower than the standard model, which takes 4.06 s to execute. This implies that additional computational complexity is introduced by Fick's Law optimization, possibly as a result of feature engineering or additional mathematical transformations.

Figure 7 depicts the execution time (in seconds) for different random forest parameters. The maximum depth of the decision trees, which ranges from 10 to 25, is represented by the x-axis.

The number of estimators, or decision trees, is represented by the y-axis and ranges from 50 to 200. In the random forest regressor model, the numbers 50, 100, 150, and 200 indicate the number of estimators (forest trees) that are used. The heatmap visualizes how the execution time varies with changes in the number of estimators and the maximum depth of the trees. The color bar on the right represents the scale of execution time in seconds, where lighter colors indicate longer times and darker colors indicate shorter times. The data suggest that as the number of estimators and maximum depth increase, the execution time also tends to increase, especially for larger values of both parameters. There is a greater effect on execution time when the number of estimators is increased because a larger ensemble necessitates training and evaluating more trees.

6 Conclusion

This paper proposes two mathematical models and a machine learning approach to accurately calculate the energy consumption of the VLOL drone. Eldosuky, the first model, examines how weight, thrust, and air–rotor interaction affect power consumption. The second model, Eman, considers additional factors like the drone's mass, payload, and environmental variables such as altitude and airspeed. Furthermore, real data are used to simulate drone performance and validate energy predictions using a random forest regressor. Three phases of motion are included into the model's outputs by the Fick's law algorithm (FLA) optimizer, which is used to improve prediction accuracy. These phases are referred to as diffusion operator (DO), equilibrium operator (EO), and steady-state operator

Table 6 The time complexity of Fick's law algorithm

Phase		Operation	Time complexity (Big O)	Explanation
Initialization phase	Initializing parameters	C1,C2,C3,C4,C5,D	O(1)	Initializing constants takes constant time
	Initializing population	Random Initialization	O(N)	Each molecule is initialized independently
Clustering phase	Dividing the population into two groups	Grouping	O(N)	Simple division into two groups, linear in population size
	Fitness computation	Fitness Calc	O(N)	Calculating fitness for each molecule (linear in population size)
	Finding the best molecule and global optimum	Search	O(N)	Requires a linear search for the best molecule
Main loop	Loop condition	Looping MAXFES times	O(MAXFES)	Iterates up to the maximum function evaluations
	Steady-state operator (SSO)	Update Position (SSO)	O(N)	Each molecule's position is updated in linear time
	Equilibrium operator (EO)	Update Position (EO)	O(N)	Each molecule's position is updated in linear time
	Diffusion operator (DO)	Update Position (DO)	O(N)	Linear update of each molecule's position
Fitness update		Fitness calculation	O(N)	Recomputing fitness for each molecule after updates
Termination and return			O(1)	Constant time to return the best solution

Table 7 Execution times for the random forest model

Model	Execution time (s)
Random forest (no optimization)	4.06
Fick's law optimized random forest	48.59

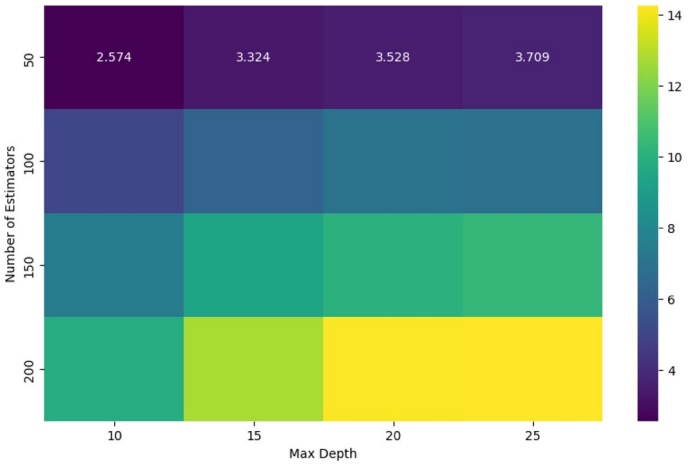

Fig. 7 Execution time for different random forest parameters

(SSO). Seven models (Eldosuky, Eman, D'Andrea, Dorling, Stolaroff, Kirchstein, and Tseng) have their evaluation metrics compared both before and after normalization. Eldosuky model has lower MAE (313.3711) and RMSE (500.7558) before normalization. Following normalization, the Fick's law algorithm optimizer exhibits optimal metrics, showcasing a noteworthy enhancement with an RMSE of 0.24303 and an MAE of 0.08805.

Author contributions Mohamed El-dosuky performs design and implementation. Data analysis, methodology, and writing the original draft are performed by Eman I. Abd El-Latif.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

Data availability The data that support the findings of this study are available from author Eman I. Abd El-Latif, upon reasonable request.

Code availability The code that supports the findings of this paper is available from author Mohamed El-dosuky, upon reasonable request.

Declarations

Conflict of interest The corresponding author certifies that there is no conflict of interest on behalf of all authors.

References

1. Agatz N, Bouman P, Schmidt M (2018) Optimization approaches for the traveling salesman problem with drone. Transp Sci 52(4):965–981
2. Cheng C, Adulyasak Y, Rousseau LM (2020) Drone routing with energy function: Formulation and exact algorithm. Transp Res Part B Methodol 139:364–387
3. Merkert R, Bushell J (2020) Managing the drone revolution: a systematic literature review into the current use of airborne drones and future strategic directions for their effective control. J Air Transp Manag 89:101929
4. Zaheer Z, Usmani A, Khan E, and Qadeer MA (2016, July) Aerial surveillance system using UAV. In: 2016 thirteenth international conference on wireless and optical communications networks (WOCN), pp 1–7. IEEE
5. Thibbotuwawa A, Nielsen P, Zbigniew B, and Bocewicz G (2019) Energy consumption in unmanned aerial vehicles: a review of energy consumption models and their relation to the UAV routing. In: Information Systems Architecture and Technology: Proceedings of 39th International Conference on Information Systems Architecture and Technology–ISAT 2018: Part II, pp 173–184. Springer International Publishing
6. Hu S, Wu Q, Wang X (2020) Energy management and trajectory optimization for UAV-enabled legitimate monitoring systems. IEEE Trans Wireless Commun 20(1):142–155
7. Zhang J, Campbell JF, Sweeney DC II, Hupman AC (2021) Energy consumption models for delivery drones: a comparison and assessment. Transp Res Part D Transp Environ 90:102668
8. Dorling K, Heinrichs J, Messier GG, Magierowski S (2016) Vehicle routing problems for drone delivery. IEEE Trans Syst Man Cybern Syst 47(1):70–85
9. Alyassi R, Khonji M, Karapetyan A, Chau SCK, Elbassioni K, Tseng CM (2022) Autonomous recharging and flight mission planning for battery-operated autonomous drones. IEEE Trans Autom Sci Eng 20(2):1034–1046
10. Prasetia AS, Wai RJ, Wen YL, Wang YK (2019) Mission-based energy consumption prediction of multirotor UAV. IEEE Access 7:33055–33063
11. Stolaroff JK, Samaras C, O'Neill ER, Lubers A, Mitchell AS, Ceperley D (2018) Energy use and life cycle greenhouse gas emissions of drones for commercial package delivery. Nat Commun 9(1):409
12. Alzaghir A, Abdellah AR, and Koucheryavy A (2021, August) Predicting energy consumption for UAV-enabled MEC using machine learning algorithm. In: International Conference on Next Generation Wired/Wireless Networking, pp 297–309. Cham: Springer International Publishing
13. Hosseini S, Fard RH (2021) Machine learning algorithms for predicting electricity consumption of buildings. Wireless Pers Commun 121:3329–3341
14. Abeywickrama HV, Jayawickrama BA, He Y, Dutkiewicz E (2018) Comprehensive energy consumption model for unmanned aerial vehicles, based on empirical studies of battery performance. IEEE access 6:58383–58394

15. Rodrigues TA, Patrikar J, Choudhry A, Feldgoise J, Arcot V, Gahlaut A, Samaras C (2021) In-flight positional and energy use data set of a DJI Matrice 100 quadcopter for small package delivery. Sci Data 8(1):155
16. Di Puglia Pugliese L, Guerriero F, Scutellá MG (2021) The last-mile delivery process with trucks and drones under uncertain energy consumption. J Optim Theory Appl 191(1):31–67
17. Kwon Y, Hwang J (2020) Mathematical modeling for flocking flight of autonomous multi-UAV system, including environmental factors. KSII Trans Internet Inf Syst 14(2):595–609
18. Sekander S, Tabassum H, Hossain E (2020) Statistical performance modeling of solar and wind-powered UAV communications. IEEE Trans Mob Comput 20(8):2686–2700
19. Raffaello DA (2014) Guest editorial can drones deliver? IEEE Trans Autom Sci Eng 11(3):647–648
20. Figliozzi MA (2017) Lifecycle modeling and assessment of unmanned aerial vehicles (drones) CO_2e emissions. Transp Res Part D: Transp Environ 57:251–261
21. Hoffmann H, Misailovic S, Sidiroglou S, Agarwal A, and Rinard M (2009) Using code perforation to improve performance, reduce energy consumption, and respond to failures
22. Langelaan JW, Schmitz S, Palacios J, and Lorenz RD (2017, March) Energetics of rotary-wing exploration of Titan. In: 2017 IEEE Aerospace Conference, pp 1–11. IEEE
23. Kirschstein T (2020) Comparison of energy demands of drone-based and ground-based parcel delivery services. Transp Res Part D Transp Environ 78:102209
24. Graw JH, Wood WT, Phrampus BJ (2021) Predicting global marine sediment density using the random forest regressor machine learning algorithm. J Geophys Res Solid Earth 126(1):e2020JB020135

Authors and Affiliations

Eman I. Abd El-Latif[1] · Mohamed El-dosuky[2,3]

✉ Eman I. Abd El-Latif
eman.mohamed@fsc.bu.edu.eg

Mohamed El-dosuky
maldosuky@arabeast.edu.sa

[1] Faculty of Science, Benha University, Benha, Egypt

[2] Computer Science Department, Arab East Colleges, Riyadh, Saudi Arabia

[3] Computer Science Department, Faculty of Computers and Information, Mansoura University, Mansoura, Egypt