State of Charge Estimation for Lithium-ion Battery Based on Random Forests Technique with Gravitational Search Algorithm

M.S. Hossain Lipu, A. Ayob, M.H.M. Saad and Aini Hussain
Faculty of Engineering and Built Environment
Universiti Kebangsaan Malaysia
Bangi 43600, Selangor, Malaysia
lipu@siswa.ukm.edu.my; afida.ayob@ukm.edu.my;
hanifsaad@ukm.edu.my; draini@ukm.edu.my;

M.A. Hannan and M. Faisal
Department of Electrical Power Engineering
Universiti Tenaga Nasional
Kajang 43000, Selangor, Malaysia
hannan@uniten.edu.my; se22694@utn.edu.my

Abstract— **An accurate state of charge (SOC) estimation for lithium-ion battery has been an intensively researched subject in electric vehicle (EV) application towards the advancement of the sustainable transportation system. However, SOC estimation with high accuracy is challenging because of the complex internal characteristics of the lithium-ion battery which is changed by different environmental situations. This paper develops an accurate method for the state of charge (SOC) estimation of a lithium-ion battery using random forests (RFs) algorithm. However, the accuracy of RFs highly depends on the appropriate selection of trees and leaves per tree in a forest. Thus, this research develops an enhanced model with RFs based gravitational search algorithm (GSA). The aim of GSA is to find the best value of trees and leaves per tree. The robustness and accuracy of the proposed model are tested under different temperatures. The model training and validation are executed using federal urban driving schedule (FUDS). The effectiveness of the proposed method is compared with the conventional RFs and radial basis function neural network (RBFNN) and optimal RBFNN-GSA models using different statistical error terms and computational cost. The proposed RFs based GSA model offers higher robustness and accuracy in reducing RMSE by 55.4%, 67.4%, and MAE by 39.1% and 78.1% than conventional RFs and RBFNN based GSA model, respectively at 25°C.**

Keywords— State of charge, Lithium-ion battery, Random forests, Gravitational search algorithm, Electric vehicle

I. INTRODUCTION

Electric vehicles (EVs) have gained a huge interest in the automobile industry due to their improved performance and the ability to reduce carbon emission [1]. However, the market of EV is still struggling to prosper due to the short travel distance of EV and the high price of energy storage systems. The lithium-ion battery has high market potential specifically in EV applications around the world due to light weight, long lifespan, low memory effect, high voltage and high energy density [2]. Nevertheless, EV run by the lithium-ion battery is facing some serious challenges such as battery lifecycle improvement, fast charging approach, charging and discharging control, safety and protection and battery parameters estimation [3].

State of Charge (SOC) of a lithium-ion battery indicates the amount of charge that is available inside a cell to run a vehicle [4]. An improved and robust SOC estimation model can be used for charge equalization in series connected lithium-ion cells for improving the battery lifecycle [5]. However, SOC cannot be evaluated in a straightforward way due to the complex electrochemical reactions. In addition, various internal and external factors degrade the battery performance. Battery aging, temperature rise, battery material degradation, hysteresis, lifecycles reduction are the main hindrances to the accurate estimation of SOC.

Several methods have already been introduced for the estimation of SOC. However, the established models have some drawbacks. Coulomb counting uses discharge current to estimate SOC. However, the method suffers from cumulative effect problem which provides inaccurate results [6]. Open circuit voltage (OCV) has a strong correlation with SOC [7]. However, OCV needs long rest time which is ineffective for online operation. Kalman filter (KF) [8], particle filter (PF) [9] and H∞ filter [10] have strong filtering technique to handle high randomness of input variables. Nevertheless, a lot of mathematical equations and relationships make them inefficient to work under different environmental conditions. Artificial neural network (ANN) [11] and Fuzzy logic (FL) [12] are robust algorithms which have demonstrated to be superior to conventional and model-based techniques. ANN and FL can estimate SOC under aging, noises and temperature variation effects. Nonetheless, ANN and FL methods require lots of computation, appropriate training data and a costly processing unit.

To overcome the above challenges, an improved random forest (RF) technique is proposed that can work effectively under an EV load profile and various temperatures conditions. RF has an efficient and robust algorithm to model a complex and non-linear system. Also, RFs do not have underfit and overfit problem, has fast estimation speed and works effectively in handling large datasets, which delivers superior performance [13]. Furthermore, RFs-based prediction is evaluated without battery model and parameters relationship. Li et al [14] estimated SOC using RF regression in steady and

978-1-5386-5686-0/18/$31.00 ©2018 IEEE

dynamic conditions. However, the study did not consider the temperature variations. Furthermore, the SOC was examined using the default value of trees (500) and leaves per tree (2) which did not provide optimal solutions. Therefore, this research has added a new contribution to enhance the RF computational capability by employing gravitational search algorithm (GSA) to search for the optimal value of trees and leaves under EV drive cycle at varying temperature conditions.

II. Random Forest Algorithm

Breiman [13] developed an improved machine learning technique called random forests (RFs) using bagging and decision trees. Bagging is a method that is used for decreasing the variance of a projected estimation function. RFs are the extension of bagging which is constructed using de-correlated trees. The RFs are modeled by picking up a group of small input dataset and then splitting them in a random order. The procedure of RFs begins with the formation of new dataset equal to the length of the original data. The bootstrapping technique is used to choose the data in a random way from the original data set. A sequence of binary splits is formed from the new dataset to create the decision trees.

A. Classification and regression

The responses to the estimated data are created using classification and regression. The response comes from the decision of each tree in the forest that generates from the root node and then transfers to a leaf node. The regression trees responses are expressed in numeric form while the classification trees responses are presented in nominal form (true or false). The following steps are followed to construct the decision trees. At first, the input variables are used to create the randomly ordered binary split of each predictor. Then, the best spilt is selected based on the optimization method. Secondly, the selected split is forced to break down into two new child nodes. Lastly, the similar procedure is also applied to new child nodes. Nevertheless, the optimization method assessment and stopping rule are required for the execution of the earlier process.

B. Regression algorithm

The prediction in RFs algorithm is performed using the training and testing process. In the training stage, RFs use the original training data to draw N multiple bootstrap samples. Then, classification or regression trees (CART) is created for each bootstrap sample. At each node of the CART, the split with the lowest error rate of the predictors is selected from the random sample. The bootstrap sample and tree growth technique are used to predict data at each bootstrap iteration. Then, the predictor error rate and aggregate OBB predictions are estimated. Evaluation of the average summation of the predictor's estimates of the new data of final node through all the trees. RFs error rate depends on the individual strength of each tree and correlation between trees. When the training set is formed from the sampling with replacement, OBB data is generated by leaving out one-third of data. The internal structure of the data and variable importance are estimated by OBB dataset.

C. Variable importance measure and cluster analysis

The individual impacts of the inputs on the output is determined by the variable importance. RFs utilize the alteration-importance measure and OBB data to estimate variable importance. The difference between the prediction accuracy before and after altering the variables (z) averaged over all the predictors result in the variable importance. The mathematical expression of variable importance can be expressed by,

$$VI^{(tr)}(z)=\frac{\sum_{T}\left(\frac{\sum_{x_i\in\beta^{c(tr)}} I\left(L_j=c_i^{(tr)}\right)}{\left|\beta^{c(tr)}\right|}-\frac{\sum_{x_i\in\beta^{c(tr)}} I\left(L_j=c_{i,\pi z}^{(tr)}\right)}{\left|\beta^{c(tr)}\right|}\right)}{T} \quad (1)$$

Where x_i is the sample value, i is sample number per leave , j is the sample number per tree in the forest, $\beta^{c(tr)}$ denotes the OBB samples, tr corresponds to tree number (1,2,…, T), L_j is the true label, I signifies the importance function depends on L_j , T represents the total number tree, $c_i^{(tr)}$ and $c_{i,\pi z}^{(tr)}$ indicates the predicted classes for a tree before and after the variable is altered, respectively.

Secondly, the outliers in the training data is found by the cluster analysis. The outlier or clustering is a process of identifying the observations which do not belong to the dataset. Variable importance and cluster method improve the prediction results significantly. The structure of RF algorithm is shown in Fig. 1.

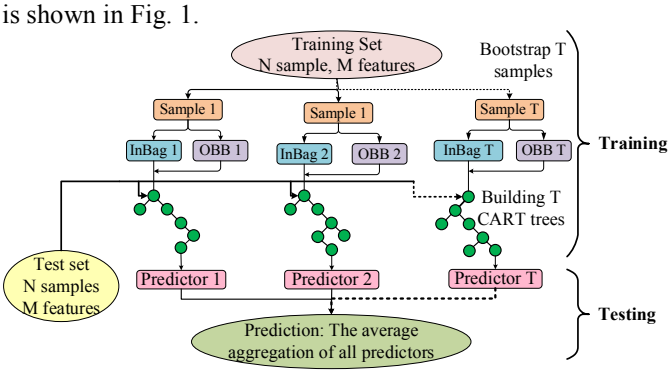

Fig 1. Flow diagram of random forest algorithm structure

III. Gravitational Search Algorithm

Rashedi et al. [15] used the principle of gravity law and mass interactions to invent a physics-based algorithm called the gravitational search algorithm (GSA). The laws of motion and the law of Newtonian gravity are used to develop GSA model, which states that "every particle in the universe attracts every other particle with a force that is directly proportional to their masses and inversely proportional to the square of the distance between them" as shown in the equations below.

$$F=G\frac{M_1M_2}{R^2} \quad (2)$$

Where F represents the gravitational force magnitude; G denotes the gravitational constant; M_1 and M_2 indicates the mass of the first and second particles, respectively; and R is measured by calculating the distance between the two

particles. Acceleration a is calculated using Newton's second law, can be expressed as,

$$a = \frac{F}{m} \tag{3}$$

Gravitational constant (t) is assessed by calculating the ratio between initial time t_0 and actual time t multiplied by gravitational constant, (t_0), as follows

$$G(t) = G(t_0) \times \left(\frac{t_0}{t}\right), \beta<1 \tag{4}$$

The locations are initialized for the N number of the initialization agents. The masses are chosen randomly within the specified search interval, as presented as follows:

$$X_i = \left(X_i^1, \ldots \ldots X_i^d, \ldots \ldots X_i^n\right), \textit{ for } i=1,2,\ldots,N \tag{5}$$

Where X_{di} is the position of *i-th* agent in the *d-th* dimension and n is the space dimension. The mathematical equations for solving problems with best and worst value and the masses of each agent, are presented as,

$$best(t) = \min_{j\in\{1,\ldots,N\}} fit_j(t) \tag{6}$$

$$Worst(t) = \max_{j\in\{1,\ldots,N\}} fit_j(t) \tag{7}$$

$$m_i(t) = \frac{fit_j(t) - Worst(t)}{best(t) - Worst(t)} \tag{8}$$

$$M_i = \frac{m_i(t)}{\sum_{j=1}^{N} m_i(t)} \tag{9}$$

The total force F for the *i-th* agent is computed using the gravitational constant G, position X, acceleration a, velocity v in different directions at the next iteration t, are as follows:

$$G(t) = G_0 e^{(-\alpha/T)} \tag{10}$$

$$F_{ij}^d(t) = G(t)\frac{M_{pi} \times M_{\alpha j}}{R_{ij} + \varepsilon}\left(X_j^d(t) - X_i^d(t)\right) \tag{11}$$

$$F_i^d(t) = \sum_{j\in Kbest, j\neq i} rand_j Fit_{ij}^d(t) \tag{12}$$

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \tag{13}$$

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{14}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{15}$$

The pseudo code of GSA is described as follows [15].

1. Search space identification, t=0
2. Randomized initialization X_i *(t)* for i=1,2,…., N
3. Fitness evaluation of agents.
4. Update G(t), Best (t), Worst the (t), M_i *(t)* for i=1,2,…., a N
5. Calculation of acceleration and velocity
6. Updating agent position to yield X_i *(t)*for i=1,2,…., N, t=t+1
7. Repeat steps 3 to 7 until the stopping criterion is reached.

IV. Data Development

This research was conducted by collecting data from the Center for Advanced Life Cycle Engineering (CALCE) battery group [16]. The test battery material includes lithium nickel manganese cobalt oxide (NMC) to examine SOC. NMC has a nominal capacity of 2.0 Ah. Battery charging was performed using constant current-constant voltage (CCCV) approach. The data was observed in each second. CALCE developed a battery test bench with Arbin BT2000, a thermal chamber, the host computer and test battery. The federal urban driving schedule (FUDS) is loaded to extract four significant input variables including current *(I),* voltage *(V),* power *(P)* and temperature *(T)* at three different temperatures (0°C, 25°C and 45°C). FUDS profile input dataset for SOC estimation is illustrated in Fig. 2.

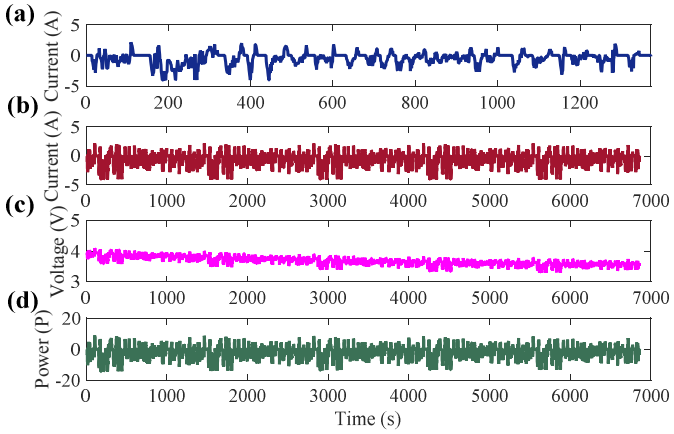

Fig. 2. FUDS drive cycle (a) current in one cycle (b) current, (c) voltage, and (d) power

V. Random Forest Based GSA Model

The estimation of the SOC starts by developing a battery test bench and then extracting data from the battery cell. Data normalization and moving average filtering methods are applied to improve the estimation speed and reduce the data fluctuation, respectively. Data normalization is expressed in (16).

$$x = \frac{2\left(x - x_{\min}\right)}{x_{\max} - x_{\min}} - 1 \tag{16}$$

Where, $x_{\max}$ and $x_{\min}$ denote the maximum and minimum value of input variables. The duration of one FUDS cycle is 1372 seconds [16]. Training and validation of model are performed using five consecutive FUDS drive cycles. RFs based SOC is evaluated using 70% selected data for training and 30% selected data for testing. After that, the optimal value of trees and leaves in RFs algorithm is found by GSA using the minimum value root means square error (RMSE). The goal of GSA optimization is to achieve high accuracy and best fitting for the prediction performance. When the iteration process of GSA is completed, the best value of trees and leaves proceed to RFs model for final training and testing. Finally, SOC is examined and the obtained value is verified with the reference value. Coulomb counting method with the appropriate tuning of the current sensor is used as a reference. The SOC performance is further examined using different statistical error terms including RMSE, mean absolute error (MAE) and mean absolute percentage error (MAPE), which are expressed as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(I_{es} - I_a\right)^2} \quad (17)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left(I_{es} - I_a\right) \quad (18)$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{I_a - I_{es}}{I_a}\right| \quad (19)$$

The flowchart of SOC estimation using RF based GSA is presented in Fig. 3.

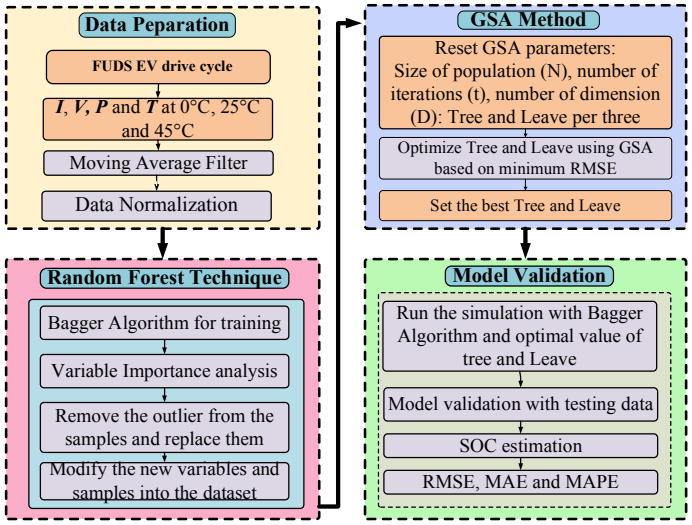

Fig. 3. Flowchart of RFs based GSA for SOC estimation

A. Random forest implementation

The RF regression algorithm begins by assigning the input vector $X = x_1, x_{2,\ldots,} x_p$ that are used to build up a forest. A set of K trees is determined by $\{T_1(x), T_2(x), \ldots, T_k(x)\}$ inside the forest. The output of RF for each tree $\hat{Y}_1 = T_1(X), \ldots, \hat{Y}_m = T_m(X)$ where m = 1, … , K. The final outcome of the RF is predicted by evaluating the mean value of predictions for all trees and can be expressed by

$$D = D_1, D_{2,\ldots,} D_n = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \quad (20)$$

The training dataset is expected to be drawn independently from the input and output $D = D_1, D_{2,\ldots,} D_n = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where x_i, i = 1, … , n represents the training dataset for the input vector and y_i, i = 1, … , n denotes the training dataset for the output vector. The methodology of RFs for regression is described as follows.

Step 1 Each regression tree in a forest is developed using a training dataset and drawn by the randomly formed bootstrap sample with replacement. Breiman suggested to use the default values which is 500 trees. In some cases, and two times default or half of the default values are used depending on application and data variance. Then, the best value is selected that delivers good results. This process is similar to trial and error approach, However, Breiman did not reveal any mathematical procedures to search for the optimal value of trees and leaves in the forest. A different pattern of the independent dataset is used for each bootstrap sample to improve the RFs model performance. The 'out-of-bag (OOB)' data is formed by removing one third data from the training dataset and 'in-Bag' data is formed using the remaining dataset.

Step 2 The binary rule is created by selecting the number of splits randomly at each node of the tree. Nevertheless, mean squared error (MSE) for each split tree is calculated and then checked with the obtained OBB data. The splits which have the best performance is chosen as a number of leaves per tree. Usually, five (5) leave per three is set as the default number and this value remains constant during the training phase.

Step 3 In this training stage, the significance of each variable is measured using (1). The accuracy of prediction increases with the decrement of variable importance.

Step 4 Then, the cluster method is used to identify the uncorrelated dataset and noise points. The density model is employed as a cluster analysis type. The efficiency of RF is improved substantially with the removal of the outlier's data from the training dataset, hence the outliers data are recognized, then separated, and substituted.

Step 5 Finally, SOC is estimated in the testing stage by computing the mean value of the predictors for all regression trees.

B. GSA implementation

RFs use the default number of trees (500) and leave (5) to predict the outcomes. However, selecting the default value does not guarantee the optimal result to SOC estimation. Therefore, GSA is utilized to address the problem. GSA helps to search for the optimal number of trees and leaves per tree. The implementation of GSA is described as follows.

Step 1 SOC estimation starts with assigning the GSA parameters. The population size (agent) and the iteration number are assigned as 100 and 250, respectively. In addition, the boundary limit of trees and leaves is selected. For trees and leave, the limit lies between '0' and '500' and '0' and '10', respectively.

Step 2 The random order agent position including trees and leaves per tree are generated within the specific boundary range.

Step 3 The bagger algorithm is used to train the data of trees and leaves.

Step 4 The evaluation of the fitness function of each agent is determined based on RMSE.

Step 5 The gravitational constant G, and best, worst position of the agent are updated using (10)-(12).

Step 6 The acceleration and velocity of the agent are calculated using (13) and (14).

Step 7 The agent position is updated using (15).

Step 8 The RMSE of each agent is re-verified using the

RFs activation function.

Step 9 The best agent location for a number of trees and leaves is chosen based on minimum RMSE achieved through all iterations.

Step 10 The final RFs training and validation are executed with an optimal value of trees and leaves.

VI. Results and Discussions

A. Objective function assessment and optimal parameters

The objective function is assessed to determine the minimum error rate after specific iterations. In this research, RMSE is chosen as the objective function. The lowest RMSE value after certain iterations provides the optimal value of trees and leaves. The optimization response curves of GSA at different temperatures are shown in Fig. 4.

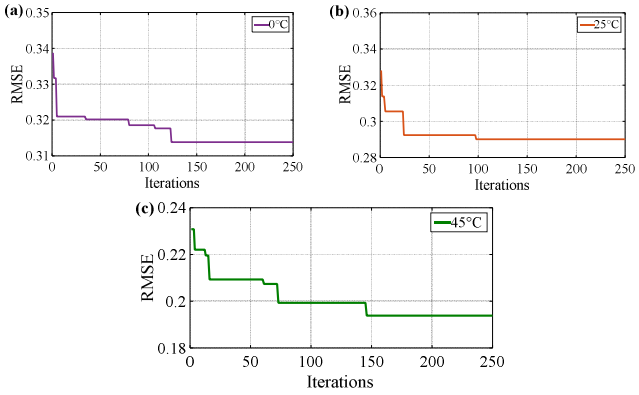

Fig. 4. Convergence characteristics curves of GSA (a) 0°C, (b) 25° and, (c) 45°C

The lowest value of RMSE is obtained after 125, 99 and 145 iterations amounting to 0.314%, 0.292% and 0.198% at 0°C, 25°C and 45°C, respectively. It is evident from Table I that the accuracy of RFs algorithm is significantly improved with the implementation of GSA. Table I shows the best values of RF at different temperatures. Temperate has a significant impact on objective function evaluation since temperature rise results in a decrease in capacity of the lithium-ion battery. When the ambient temperature raises, electrolyte activity increases and viscosity decrease which causes ion diffusion.

TABLE I. Optimal values of RF at Different Temperatures

Temperature	0°C	25°	45°C
Lowest value of RMSE	0.314%	0.292%	0.198%
No. of Tree	462	343	444
No. of Leave	4	3	1

B. SOC Estimation

The SOC estimation results based on RFs with GSA under the FUDS drive cycle at 0°C, 25°C and 45°C is illustrated in Fig. 5(a), 5(c) and 5(e). The estimation results are compared with radial basis function neural network (RBFNN) to prove the dominance of the proposed model. RBFNN algorithm is used in common as a machine learning technique to estimate SOC. To perform a fair comparison, the center, width and hidden neurons of RBFNN are optimized using GSA. The comparison has been analyzed using similar training and testing dataset, training function, population size and iteration number. It is noticed in Fig. 5 that SOC estimation based on RFs with GSA is nearly aligned to the reference value whereas, in RBFNN based GSA model, the SOC line has high oscillations at varying temperature conditions.

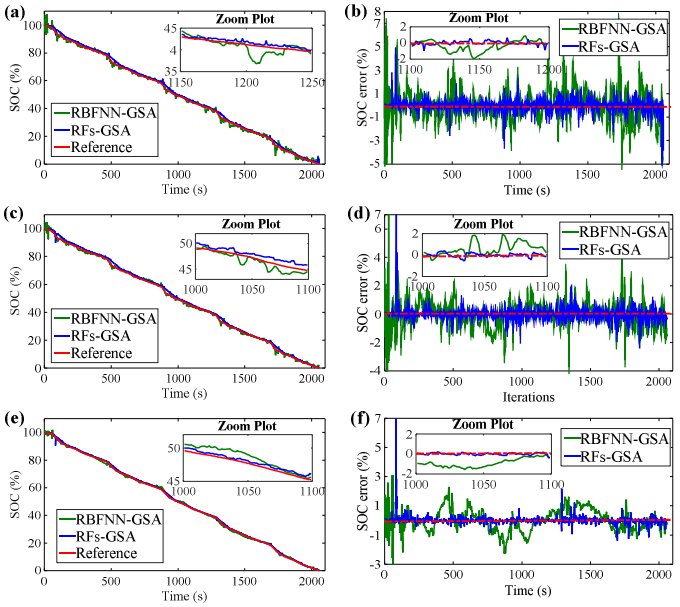

Fig. 5. SOC at (a) 0°C, (c) 25°C, (e) 45°C; SOC error at (b) 0°C, (d) 25°C, (f) 45°C in FUDS EV drive cycle

The accuracy of SOC estimation results is also examined using the error rate which is found by calculating the difference between the estimated SOC and actual SOC, as presented in Fig. 5(b), 5(d) and 5(f). RBFNN-GSA model obtains SOC with high fluctuations and has some random peaks at certain intervals. On the contrary, the developed RFs-GSA achieves more stable results and has few fluctuations. In RBFNN-GSA model, SOC error is attained at between -8.2% and 6.7% at 0°C. Nonetheless, in RFs-GSA model, SOC error is low and found between -4.5% and 4.3%. As the temperatures increase from 25°C to 45°C. the error bounds in RBFNN-GSA model reduces and lies between -3.4% to 3.1%. Nevertheless, steady operation and less error are observed in RFs based GSA model. The major percentage of error is noticed between -1.4% and 2.2% at 45°C expect some high peaks during initial SOC estimation. The results demonstrate that RFs based GSA model offers better accuracy and robustness than the RBFNN based GSA model.

C. Performance assessment comparison

The effectiveness of SOC estimation based on RFs with GSA is validated using RMSE, MAE, MAPE and computation time (s) in the FUDS cycle at different temperature conditions. A detailed comparison of the proposed method and RBFNN, RBFNN-GSA and RFs are analyzed, as presented in Table II. The hyperparameters of conventional RBFNN and RFs are determined using trial and error method. RFs-GSA obtains RMSE of 0.29% at 25°C which is a 70.4%, 67.4% and 55.4% decrease from the RBFNN, RBFNN-GSA, and RFs models, respectively. The accuracy of RFs-GSA also increases with

the rise of temperatures and achieves lower RMSE of 0.22% at 45°C which is a 74.1 %, 73.6% and 62.1% reduction from the RBFNN, RBFNN-GSA and RFs models, respectively. Moreover, RFs-GSA model achieves MAE of 0.14% at 25°C which is reduced by 81.5%, 78.1%, and 39.1% compared to RBFNN, RBFNN-GSA, and RFs model, respectively. In addition, MAPE is improved in proposed model and is dropped by 74%, 66.1% and 25.7% at 25°C in comparison to the RBFNN, RBFNN-GSA and RFs model, respectively.

TABLE II. PERFORMANCE COMPARISON OF SOC ESTIMATION

Model	Temperature	RMSE (%)	MAE (%)	MAPE (%)	Computation time (s)
RBFNN	0°C	1.37	1.12	16.35	81.83
	25°C	0.98	0.76	10.83	78.92
	45°C	0.85	0.67	7.92	77.32
RBFNN-GSA	0°C	1.28	0.92	13.68	74.35
	25°C	0.89	0.64	8.29	72.65
	45°C	0.76	0.58	4.14	68.28
RFs	0°C	0.91	0.33	10.08	8.97
	25°C	0.65	0.23	3.78	8.38
	45°C	0.58	0.21	2.82	7.98
RFs-GSA (Proposed Model)	0°C	0.33	0.18	7.26	7.47
	25°C	0.29	0.14	2.81	7.24
	45°C	0.22	0.12	1.53	6.94

RFs-GSA is also dominant to other models in terms of computation speed. The computation time is the duration which is needed for model training and validation. Table II also reveals the comparison of computation time among four models. As expected, the estimation speed of RFs-GSA is higher than other three models. For example, RBFNN-GSA estimates SOC in 72.65 seconds at 25°C. Nevertheless, RFs-GSA completes training and validation in only 7.24 seconds. In conclusion, the high accuracy and fast convergence speed are achieved using RFs-GSA model. The results demonstrate that the developed model is accurate and can operate in varying environmental conditions.

VII. CONCLUSION

An enhanced SOC estimation model for a lithium-ion battery is developed using RFs based GSA. Generally, the RFs parameter selection are performed by trial and error method or by experience which consumes a lot of time and delivers inaccurate results. Hence, GSA is employed to enhance the estimation accuracy of RFs by selecting the appropriate value of trees and leaves. The model robustness is checked under the FUDS drive profile at three different temperatures. The main contributions of this research are summarized as follows: (1) the implementation of GSA not only avoids ineffective the trial and error approach but also improves the intelligence capability of RFs. (2) The proposed model estimates SOC without battery model and parameters, (3) The RFs model provides satisfactory solutions under dynamic load profiles and temperature variations, (4) the RFs-GSA model outperforms RFs, RBFNN, and RBFNN-GSA methods, (5) the proposed method also superior to existing models in terms of convergence speed. Our future research is to develop the prototype hardware in modularized design for the real-time EV applications.

ACKNOWLEDGMENT

This research was supported by Universiti Kebangsaan Malaysia under Grant Nos. FRGS/1/2017/TK04/UKM/02/12 and DIP-2015-012.

REFERENCES

[1] M. S. H. Lipu, M. A. Hannan, A. Hussain, M. H. M. Saad, A. Ayob, and F. Blaabjerg, "State of Charge Estimation for Lithium-ion Battery Using Recurrent NARX Neural Network Model Based Lighting Search Algorithm," *IEEE Access*, vol. 6, no. 1, pp. 28150–28161, 2018.

[2] M. A. Hannan, M. S. H. Lipu, A. Hussain, and A. Mohamed, "A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations," *Renew. Sustain. Energy Rev.*, vol. 78, pp. 834–854, 2017.

[3] M. S. Hossain Lipu, M. A. Hannan, A. Hussain, and M. H. M. Saad, "Optimal BP neural network algorithm for state of charge estimation of lithium-ion battery using PSO with PCA feature selection," *J. Renew. Sustain. Energy*, vol. 9, no. 6, p. 064102, Nov. 2017.

[4] M. Cacciato, G. Nobile, G. Scarcella, and G. Scelba, "Real-Time Model-Based Estimation of SOC and SOH for Energy Storage Systems," *IEEE Trans. Power Electron.*, vol. 32, no. 1, pp. 794–803, Jan. 2017.

[5] M. Hannan, M. Hoque, P. Ker, R. Begum, and A. Mohamed, "Charge Equalization Controller Algorithm for Series-Connected Lithium-Ion Battery Storage Systems: Modeling and Applications," *Energies*, vol. 10, no. 9, p. 1390, Sep. 2017.

[6] A. Nugroho, E. Rijanto, F. D. Wijaya, and P. Nugroho, "Battery state of charge estimation by using a combination of Coulomb Counting and dynamic model with adjusted gain," in *2015 International Conference on Sustainable Energy Engineering and Application (ICSEEA)*, 2015, pp. 54–58.

[7] X. Dang, L. Yan, K. Xu, X. Wu, H. Jiang, and H. Sun, "Open-Circuit Voltage-Based State of Charge Estimation of Lithium-ion Battery Using Dual Neural Network Fusion Battery Model," *Electrochim. Acta*, vol. 188, pp. 356–366, Jan. 2016.

[8] K.-T. Lee, M.-J. Dai, and C.-C. Chuang, "Temperature-Compensated Model for Lithium-Ion Polymer Batteries With Extended Kalman Filter State-of-Charge Estimation for an Implantable Charger," *IEEE Trans. Ind. Electron.*, vol. 65, no. 1, pp. 589–596, Jan. 2018.

[9] R. Xiong, Y. Zhang, H. He, X. Zhou, and M. G. Pecht, "A Double-Scale, Particle-Filtering, Energy State Prediction Algorithm for Lithium-Ion Batteries," *IEEE Trans. Ind. Electron.*, vol. 65, no. 2, pp. 1526–1538, Feb. 2018.

[10] Q. Yu, R. Xiong, C. Lin, W. Shen, and J. Deng, "Lithium-Ion Battery Parameters and State-of-Charge Joint Estimation Based on H-Infinity and Unscented Kalman Filters," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 8693–8701, Oct. 2017.

[11] M. S. H. Lipu, M. A. Hannan, and A. Hussain, "Feature Selection and Optimal Neural Network Algorithm for the State of Charge Estimation of Lithium-ion Battery for Electric Vehicle Application," *Int. J. Renew. Energy Res.*, vol. 7, no. 4, pp. 1700–1708, Dec. 2017.

[12] J. Chen, C. Xu, C. Wu, and W. Xu, "Adaptive Fuzzy Logic Control of Fuel-Cell-Battery Hybrid Systems for Electric Vehicles," *IEEE Trans. Ind. Informatics*, vol. 14, no. 1, pp. 292–300, Jan. 2018.

[13] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[14] C. Li, Z. Chen, J. Cui, Y. Wang, and F. Zou, "The Lithium-ion Battery State-of-Charge Estimation using Random Forest Regression," no. 1, pp. 336–339, 2014.

[15] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," *Inf. Sci. (Ny).*, vol. 179, no. 13, pp. 2232–2248, Jun. 2009.

[16] "CALCE, Lithium-ion battery experimental data [online]. Available: http://www.calce.umd.edu/batteries/data.htm. Accessed on: Januray 05, 2017." .