Neural Computing and Applications (2020) 32:15191–15207
https://doi.org/10.1007/s00521-020-04871-1

ORIGINAL ARTICLE

Formula-E race strategy development using artificial neural networks and Monte Carlo tree search

Xuze Liu[1] · Abbas Fotouhi[1]

Received: 13 March 2019 / Accepted: 19 March 2020 / Published online: 30 March 2020
© The Author(s) 2020

Abstract
Energy management has been one of the most important parts in electric race strategies since the Fédération Internationale de l'Automobile Formula-E championships were launched in 2014. Since that time, a number of unfavorable race finishes have been witnessed due to poor energy management. Previous researches have been focused on managing the power flow between different energy sources or different energy consumers based on a fixed cycle. However, there is no published work in the literature about energy management of a full electric racing car on repeated course but with changeable settings and driving styles. Different from traditional energy management problems, the electric race strategy is more of a multi-stage decision-making problem which has a very large scale. Meanwhile, this is a time-critical task in motorsport where fast prediction tools are needed and decisions have to be made in seconds to benefit the final outcome of the race. In this study, the use of artificial neural networks (ANN) and tree search techniques is investigated as an approach to solve such a large-scale problem. ANN prediction models are developed to replace the traditional lap time simulation as a much faster performance prediction tool. Implementation of Monte Carlo tree search based on the proposed ANN fast prediction models has provided decent capability to generate decision-making solution for both pre-race planning and in-race reaction to unexpected scenarios.

Keywords Formula-E race strategy · Energy management · Machine learning · Artificial neural networks

1 Introduction

1.1 Formula-E racing

The first worldwide electric motorsport event, the Fédération Internationale de l'Automobile (FIA) Formula-E (FE) championship, was originally conceived as a single-seater electric motor racing championship in 2011 and launched its first ePrix in September 2014 [1]. Since then, several electric racing series have been developed such as I-PACE Trophy, Electric GT. So far in any of these series, teams have been running very similar cars fitted with the same rechargeable energy storage system (RESS). It has always been a problem for engineers and drivers to properly manage the battery energy during the race. For example, a flat battery or an over-heated battery will lead to a Did Not Finish (DNF) before crossing the finished line.

Among all the series, the current fifth FIA FE season 2018/2019 has raised the problem of energy management to a completely different level. The new FE Gen 2 cars have to last for the entire 45 min plus one-lap race, and a new power mode called the 'Attack mode' is introduced into the race. Teams have to decide which two laps to activate this mode which allows teams to increase the maximum total power of the RESS by 50 kW in addition to the base 200 kW limit stated by the technical regulations.

During the race, drivers are not able to drive flat out through the complete length of the race simply due to the limited capacity of the RESS. And the integrated battery management system (BMS) also brings restrictions. When the battery temperature rises up to the first threshold, no regeneration will be allowed. If the battery temperature further reaches a higher threshold, the power output will be completely shut down, resulting in a car stopping on the

✉ Abbas Fotouhi
a.fotouhi@cranfield.ac.uk

[1] Advanced Vehicle Engineering Centre, School of Aerospace, Transport and Manufacturing (SATM), Cranfield University, Cranfield, Bedfordshire MK43 0AL, UK

track. For example, Fig. 1 demonstrates the effect of BMS in a case where we have 'No Regen' threshold of 50 °C and 'Shut Down' threshold of 60 °C during an ePrix. Figure 1a shows a typical state of charge (SOC) trend during a race. A significant steeper drop can be observed starting from 2250 s. The cause of this can be found in Fig. 1b where the battery temperature went over 50 °C at the same time. The energy consumption became quicker after then as a result of no more regeneration. If the strategy is not properly managed, it is very likely that the battery could go flat before the end of the race like shown in Fig. 1a and the battery temperature could end up over 60 °C as shown in Fig. 1b, either of which would lead to a DNF.

According to the technical regulations, there are not too many mechanical changes a driver can make once the race has started. However, drivers are free to change the powertrain settings through the steering wheel according to the regulations to help them manage the energy during the race [2]. There are three main rotary switches located at the bottom of the steering wheel by which the driver can tune the regenerate power, lift/coasting torque and drive power. Apart from these three switches, the driver can decide how much coasting distance he needs. In FE races, the regeneration and lift/coasting techniques are very popular for energy management. However, the combinations have to be properly planed as higher regeneration saves more energy but also can bring quicker temperature rise to the battery. If the driver loses his regeneration too early due to the battery temperature threshold, he will be facing an energy crisis for the remaining laps of the race. It is necessary for engineers and drivers to make careful decisions both before and during the race to avoid risking any potential DNFs.

1.2 Energy management strategy

Researches into energy management have been mainly focused on hybrid vehicles. Early researches have been trying to categorize different drive modes regarding different drive demands [3–8]. Such analytic methods have been used to achieve better efficiency on different road types, but the result in reality could be far from optimum due to the lack of analysis on demand varieties [9]. To solve such problems, more intelligent control algorithms such as fuzzy logic [10, 11] and model predictive control [12] are used to more efficiently control the power flow based on the system dynamic feedback. This feature has benefitted real-time implementations.

Later, researches indicated that the control strategies above might also not give the optimum solution as those algorithms do not take the entire trip into account. To make further global improvement, dynamic programming (DP) was proposed to manage the energy for an entire trip [13]. Machine learning has been proposed to help categorize road types and predict traffic congestion level [14–17]. And techniques such as support vector machine (SVM) [18, 19] and artificial neural networks (ANN) [9, 20] have been applied for driving condition recognition to further reduce the computation time of DP.

In addition to hybrid vehicles, full electric vehicles with multiple consumers are another object in energy management researches. These researches have focused on optimizing the power distribution between two or more motors based on their motor efficiency maps [21–23]. Techniques such as DP, genetic algorithms (GA), particle swarm optimization (PSO) are used to optimize the power distribution to minimize the power loss [24, 25]. The power distribution is also integrated with vehicle dynamics controls as a multi-objective optimization problem in recent researches [26, 27].

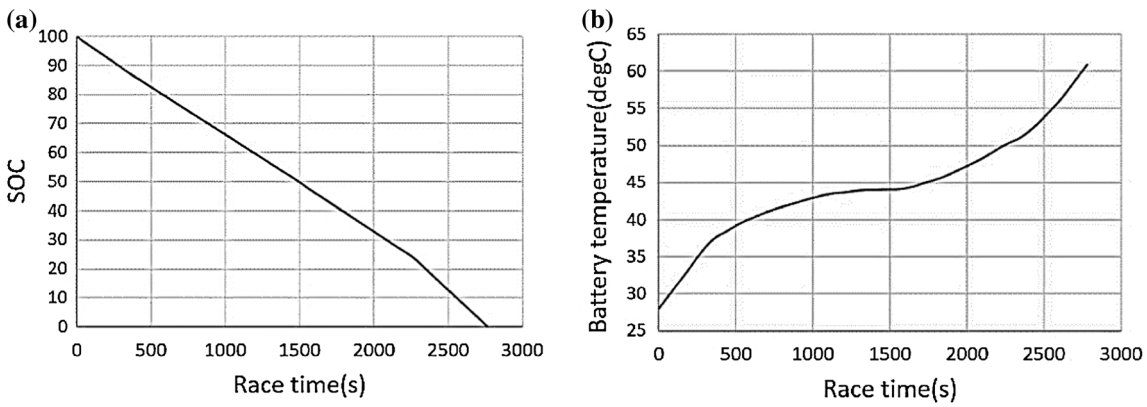

Fig. 1 **a** SOC drop in an ePrix, **b** battery temperature in an ePrix

1.3 Artificial neural networks (ANN) and Monte Carlo tree search (MCTS)

In previous researches, soft computing techniques are widely adopted to solve real-life problems in different fields. Machine learning has become a major part as a popular approach.

ANN is an important component in machine learning which was proposed as a concept in the 1940s [28]. Later in the early 1990s, the development of ANN reached a bottleneck where one-hidden-layered neural networks could not provide acceptable level of accuracy and training multi-layered networks was nearly impossible [29, 30]. After a breakthrough was made in 2006 [31], the transition from shallow networks to deep networks enabled deep learning method [32]. The training of a network may take relatively longer time due to the complexity of training algorithms, but applying a trained network is a lot quicker as beneath the network is basically matrix calculation. This made ANN models thrive and become one of the most widely used classification/prediction/fitting tools.

One of the most well-known products of machine learning in recent years is AlphaGo. What made the AlphaGo to successfully handle a problem with over 250^{150} possibilities is the combination of tree search and deep neural networks [33]. The Monte Carlo tree search (MCTS) [34] used in AlphaGo is a tree search method for finding optimal decisions based on random sampling in a given domain [35]. By focusing only on relatively more promising branches, MCTS is able to shrink a large-scale problem into a computable scale. The process of a node expanded into a next layer can be seen as a newly reached state after a decision is made. These features made the MCTS a method very suitable for large-scale multi-stage problems [36].

1.4 Present study motivation

Engineers and drivers need to be able to properly decide the race strategy both in pre-race planning and also during a race when reacting to unexpected scenarios in order to finish the race as quickly as possible. It would be easier to decide the strategy planning before the race weekend as there is much more time available. However, during the race when decisions have to be made within seconds in reaction to unexpected incidents, it would become impossible to generate a decent solution by using traditional lap time simulation approaches. The traditional lap time simulations take seconds to simulate a single lap based on a setting. To make the challenge more difficult, the number of possible settings can easily reach over 50 meaning that a 32-lap race will create a problem with a scale of 50^{32}. This is way beyond the realm of computation even before the race weekend, not to mention making quick reactions during the race. Furthermore, the race strategy development is a multi-stage decision problem, which is different from traditional energy management problem which can be solved by algorithms such as dynamic programing, linear programing. All the challenges above raise the demand for a tool with fast prediction and efficient searching capability to tackle the challenges. The present study will focus on using ANNs and Monte Carlo tree search to develop such a tool which can help engineers and drivers to make good and quick decisions in seconds.

2 Lap time simulation model and data generation

2.1 Lap time simulation

To generate accurate data to be used for training of the ANN prediction models, a lap time co-simulation platform was built by integrating IPG/Carmaker with a MATLAB/Simulink model. Different drive power, regenerate power, lift/coasting distance and torque, and environment temperature changes were simulated to study their effects on the vehicle's performance such as lap time, battery state of charge (SOC) and battery temperature. The general structure of the proposed lap time simulation platform is shown in Fig. 2. The IPG/Carmaker part comprises a FE style vehicle model, a London ePrix track model and a built-in driver model. The built-in driver model maximizes the vehicle dynamic performance on the track to deliver the fastest lap time based on the powertrain motor output torque which is calculated in the MATLAB/Simulink part according to the driver model demand and the current powertrain status, such as the battery temperature, SOC. The visualized track and lift/coasting distance definitions are shown in Fig. 3.

The function of the MATLAB/Simulink components is to provide the output torque information back to IPG/Carmaker for vehicle dynamics simulation. In this part of simulation, the battery and powertrain status are calculated based on which, given the driver demand from IPG/Carmaker, the actual regenerative/drive torque is transmitted back to IPG/Carmaker. In the MATLAB/Simulink part, the

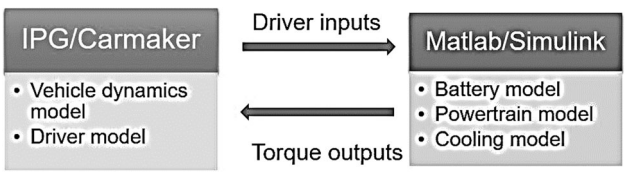

Fig. 2 Lap time simulation platform

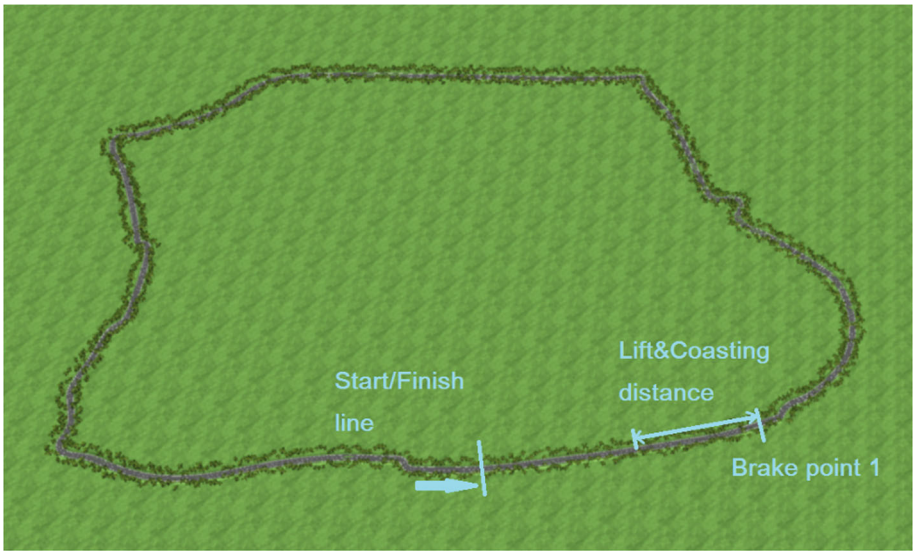

Fig. 3 London ePrix track modeling using IPG/Carmaker

electric powertrain system is modeled including a battery model which is of vital importance to predict both power limitations and heat generation. In terms of the modeling approaches used, modern battery models are mainly categorized as mathematical models, electrochemical models and electrical equivalent models [37–40]. In this application, the battery thermal behavior is the main concern. It has been widely observed in electric racing events that regenerative braking which saves energy during the race would also heat up the battery much more significantly than discharge/driving condition. Therefore, to generate reliable data samples for further decision makings, the battery model has to be able to describe such features. Considering the data available for model validation and the computing complexity of the models, Bernardi model [41] is selected in this study. Bernardi model has been widely adopted in battery thermal management studies [42–44] due to its capability to capture the reversible heat generation which differs the battery charging thermal behavior from discharging. The heat generation is calculated by the following equations:

$$Q = Q_{\text{irrev}} + Q_{\text{rev}} \tag{1}$$

$$Q_{\text{irrev}} = I^2 R \tag{2}$$

$$Q_{\text{rev}} = IT\left(\frac{\partial U}{\partial T}\right)_P \tag{3}$$

where Q is the total heat generation rate and Q_{irrev} is the irreversible heat component determined by the internal resistance of the battery and the current through it. Q_{rev} is the reversible heat component determined by the battery current, temperature and the entropy coefficient $\left(\frac{\partial U}{\partial T}\right)_P$.

In order to calculate the heat exchange between the hot coolant and ambient, the following equation is used:

$$H_{\text{real}} = \frac{A_{\text{realcooler}}}{A_{\text{refcooler}}} \cdot \frac{T_{\text{coolant}} - T_{\text{amb}}}{\Delta T_{\text{ref}}} \cdot f(H_{\text{ref}}) \tag{4}$$

where H_{real} is the heat exchange rate, $A_{\text{realcooler}}$ is the cooler size, T_{coolant} and T_{amb} are the temperatures of the coolant and air, respectively. $f(H_{\text{ref}})$ is the reference heat exchange rate look-up table at reference cooler size $A_{\text{refcooler}}$ and reference temperature difference ΔT_{ref} whose inputs are vehicle speed and coolant mass flow.

In this study, the aim is to develop the energy management strategy for a 32-lap race. Through the lap time simulation, a 32-lap race was simulated. In the race simulation, the power limit is set to 200 kW and no coasting is performed through the whole race. Issues were found at the end of the race as shown in Fig. 4. It can be observed from Fig. 4 that the battery temperature has increased above 60 °C before the end of the race and the battery went flat as well. Both these two plots indicate a DNF and that less aggressive strategy is necessary to secure a successful finish of the race. The solution of this issue is discussed in the next section.

2.2 Training data generation

The lap time simulation is able to give the energy consumption, battery temperature rise and lap time results according to the powertrain settings and environment conditions. In order to develop ANN prediction models as powerful and accurate as possible, factors which mainly determine the vehicle performance are defined as simulation inputs as listed in Table 1. There are other variables in the real world which affects the performance such as humidity, ambient pressure. However, they are less relevant and make little difference to the result and thus are not considered in this study. After these input values are

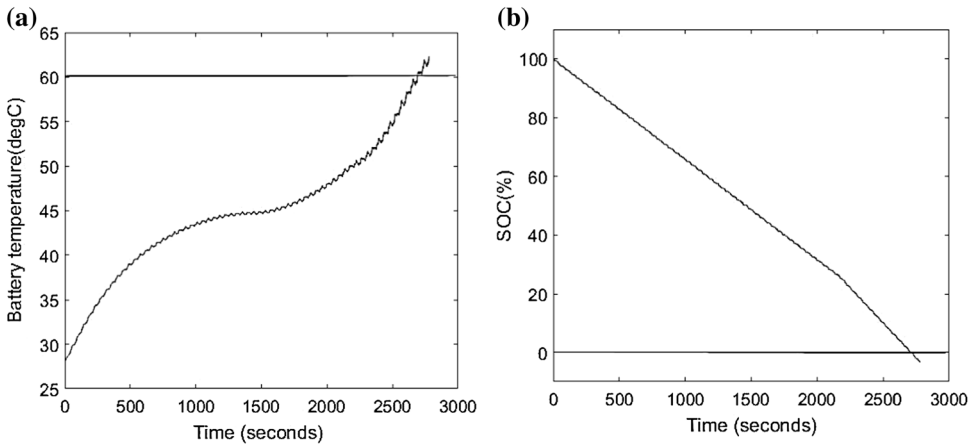

Fig. 4 Battery parameters variation during a race **a** temperature **b** SOC

Table 1 Simulation input parameters

Inputs	Descriptions	Range
Drive power limit	Maximum total power going out of the RESS	190–225 kW
Regeneration power limit	Maximum total power going into the RESS	0–250 kW
Coasting distance	Lift and coasting distance before brake point 1	0–120 m
Coasting torque	Regenerative brake torque during lift and coasting	0–200 Nm
Ambient temperature	Cooling air intake temperature	25–30 °C
Battery initial temperature	Battery temperature at the start of simulation	25–60 °C
Battery initial SOC	Battery SOC at the start of simulation	0–100%

assigned, the lap time simulation is completed for a 1-lap simulation based on these inputs. The battery temperature rise, SOC drop and lap time results are then collected as respective outputs of the inputs. In this way, each training sample comprises 7 input values and 3 output values.

In this study, the ANN prediction models function mainly as a fitting tool considering the fact that ANN is inherently good at interpolation rather than extrapolation [45]. The discrete training inputs must cover the complete possible range to guarantee well-functioning prediction networks under any circumstances. Therefore, each input is assigned with certain values instead of random assignment. Table 2 shows the values of each input and the output variables and their expected range. The sample generation was automatically completed by running a MATLAB script. The whole process took more than 3 days, and 172,800 training samples were collected for ANN prediction model development.

Table 2 Input and output values

Inputs	Discrete values
Drive power limit (kW)	190, 195, 200, 225
Regeneration power limit (kW)	0, 100, 250
Coasting distance (m)	0, 60, 120
Coasting torque (m)	0, 100, 200
Ambient temperature (°C)	25, 27, 28, 30
Battery initial temperature (°C)	25, 28.5, 31,…, 60
Battery initial SOC (%)	2.5, 5, 7.5,…, 100
Outputs	Expected range
Battery temperature rise (°C)	− 10…10
SOC drop (%)	0…5
Lap time (s)	83…87

3 Race simulation prediction using artificial neural networks

3.1 ANN layout

In this study, the number of network outputs needs to be first selected. The prediction models are expected to produce three performance values. One of the options is to use a single deep network to predict these three values simultaneously. This means that the feature extraction layers and mapping layers are shared by three different outputs. During training process, the accuracy of each neural weight will be compromised by the training algorithm in order to achieve overall optimum for three outputs, which will result in an overall accuracy compromise. In contrast, another option is to use three separate deep networks for predicting each value. In this way, each network

will only need to focus on a single output. Therefore, the accuracy of each output can be further improved. The accuracy of these two options is shown in Fig. 5. By comparing parts (a) and (b), it can be seen that the 3-output network produced twice as much mean square error as the separate network option. This result shows that separate networks are better in predicting these three values. The error difference proves that when the feature extraction and mapping layers are shared by three less-related prediction targets, the overall accuracy of the network will be compromised. Therefore, separate networks are used in this study.

The number of hidden layers, which distinguishes a 'deep network' from a 'shallow network,' is another key factor of prediction accuracy. Deep networks have stronger capabilities to deal with complex, strong nonlinear problems. In order to study the effect of network depth, networks with different layers are tested, with number of hidden neuron fixed to 10 for each hidden layer and sigmoid chosen as activation function. Three different training methods are tested and compared. Levenberg–Marquardt (LM) [46] method is used to train the SOC and lap time prediction networks, while Bayesian regularization (BR) [47] method is used to train the battery temperature network. The reason of such choice of training method is demonstrated in Fig. 6a. In terms of the mean square error after training, it can be seen that the neural networks are very good at lap time predictions and have relatively less accuracy on battery temperature predictions. Among the three training algorithms, Levenberg–Marquardt-based methods are better than the scaled conjugate gradient method. While LM produces better results in lap time and SOC predictions, BR performs better in training battery temperature prediction networks. LM is more accurate as is observed in lap time network and SOC network trainings. But when the samples are noisier like in the battery temperature case, BR performs better than the traditional LM. Therefore, when later deciding the layout of the networks, LM will be used to train the lap time and SOC networks, while BR will be used for battery temperature network as the battery temperature data are noisier than the other two.

The corresponding MSE results for different network depths are also shown in Fig. 6. It can be seen from the results that for all the three prediction objects, the mean square error decreases as the neural networks become deeper. The MSE decreases quickly before the number of layers increases to 3. More layers allow the network to extract and map higher nonlinear features from the datasets and therefore result in higher fitting quality. After the number of layers goes higher 4, network accuracy improves very slowly which suggest an adequate depth for feature extraction and mapping. By comparing the absolute error of the battery temperature prediction of different network depth (Table 3), the effect of this parameter can be seen more clearly. The deeper network produces much less errors than shallower networks. With only 0.04% of the prediction result having error greater than 0.2 °C, the 5-layer solution looks promising. Using that 5-layer network, the SOC and lap time prediction results are also very accurate. So, three 5-layer prediction networks are used in this study in the following parts.

3.2 Race prediction validation

The neural networks are developed for a single-lap performance prediction. To check the feasibility of a multi-lap race prediction using the proposed networks, a 32-lap race is predicted through 32 iterations of the three networks; meanwhile, the same race is simulated using the previously developed lap time simulation platform. The results

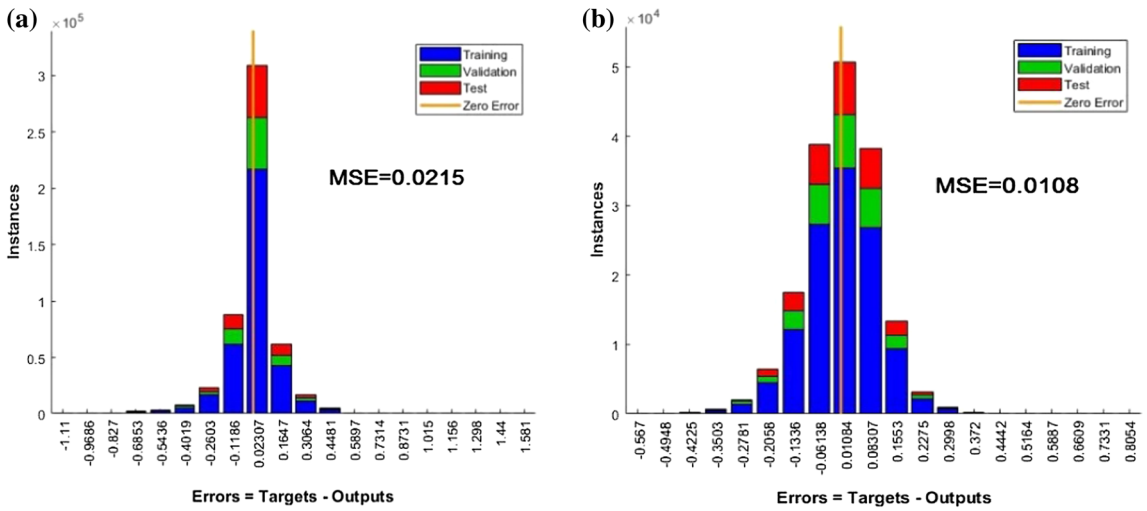

Fig. 5 **a** 3-Output network error, **b** 1-output network error

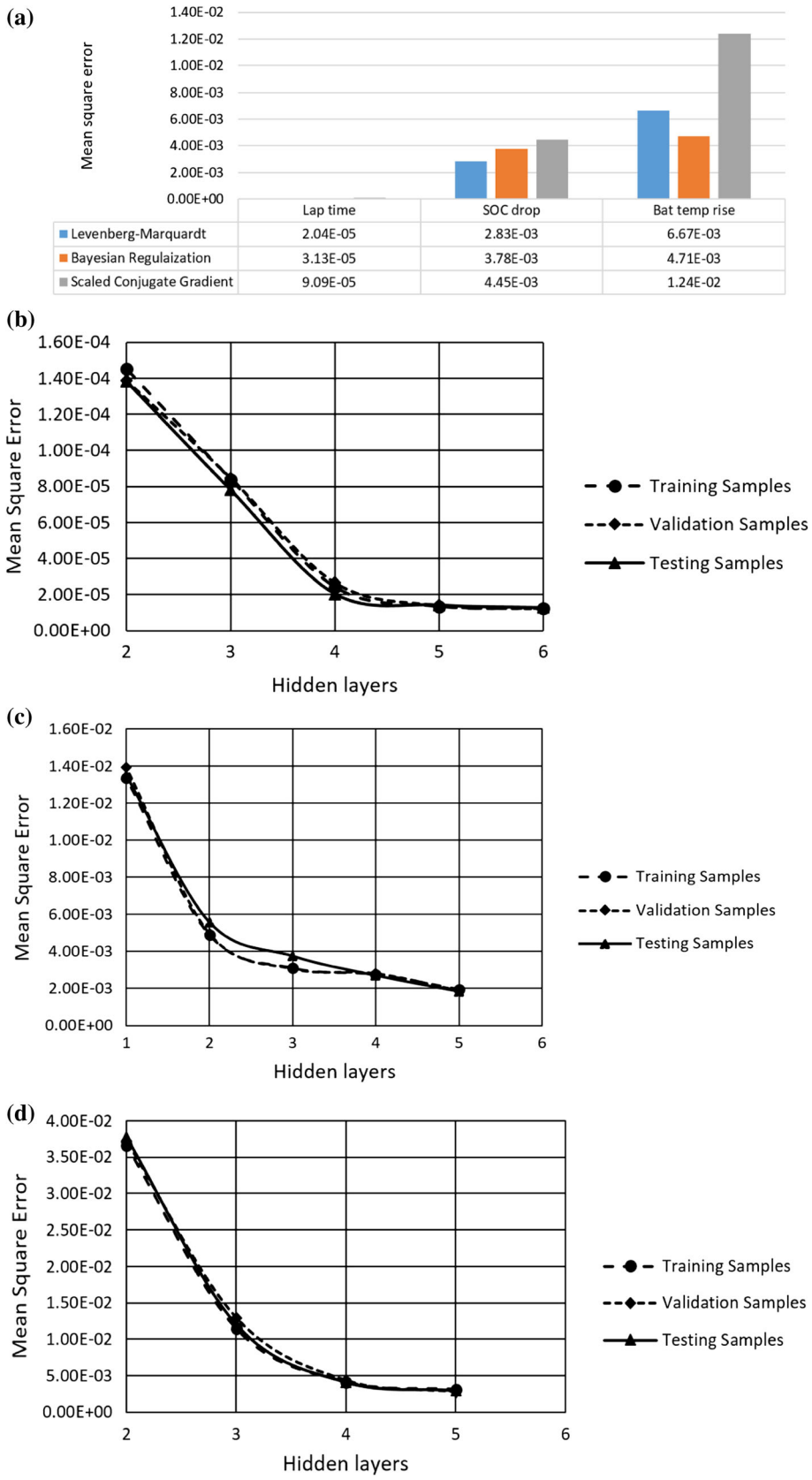

Fig. 6 MSE result for **a** different training methods **b** lap time, **c** battery SOC drop and **d** battery temperature rise

Table 3 Battery temperature prediction error comparison

	2-Layer network	5-Layer network
Biggest error	1.602 °C	0.2245 °C
Proportion of samples error > 0.4 °C	7.9%	0%
Proportion of samples error > 0.2 °C	30.9%	0.04%
Proportion of samples error < 0.1 °C	47.6%	95.6%

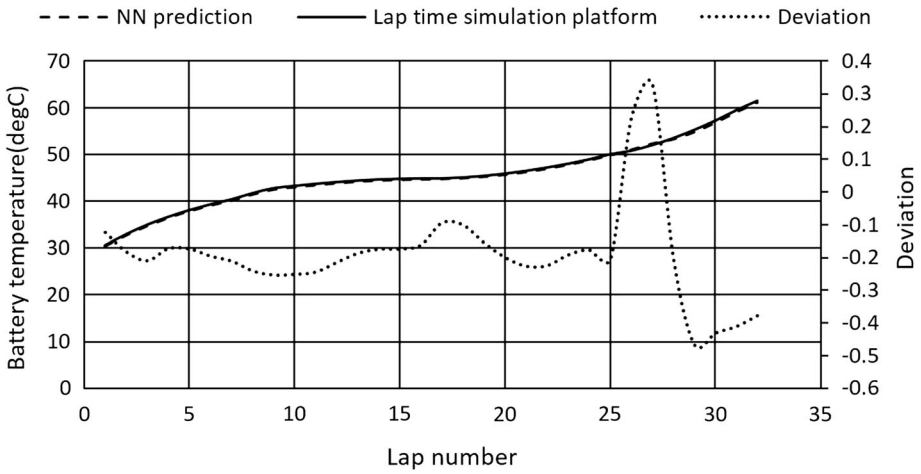

Fig. 7 Battery temperature trend throughout a race

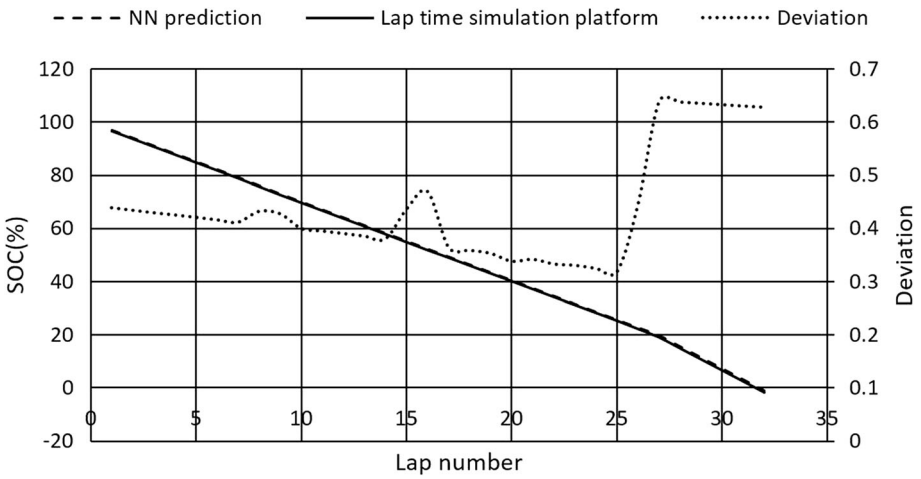

Fig. 8 SOC trend throughout a race

comparison is shown in Figs. 7 and 8. According to these results, the general patterns of ANN and simulation platform results are very close to each other. The max deviation occurred in the final laps of the race. The ANN predicted a battery temperature which is 0.37 °C lower than the lap time simulation by the end of the race.

In terms of the energy consumption, the proposed ANN predicts a very close result to the lap time simulation software too. Similar to the battery temperature, the deviation has also increased during the final laps. By the end of the race, the proposed ANN has predicted a 0.628% lower SOC than the actual simulation.

It can be seen from Figs. 7 and 8 that the prediction deviation increased after lap 26 when there is no more regeneration due to the battery high temperature (higher than 50 °C). This limit crossing results in a much faster energy consumption and thus a quicker SOC drop. Meanwhile, the missing of regeneration removed an important part of heat generation. Both these phenomenons lead to a stronger nonlinear transition of the trend of the performance indicators. Because the mechanism nature of ANN training aims to reduce the overall loss of prediction errors, this particular point (50 °C) is compromised compared to other points. As a result, relatively larger deviations can be observed in the figures mentioned above. However, the

accumulated deviation by the end of the race is very small and acceptable. In terms of prediction speed, the ANN prediction models help to save a huge amount of time. While the lap time simulation platform takes more than 20 s to simulate a single lap, within the same amount of time, the ANN can produce more than 300,000 results. It can be confirmed that the ANN prediction models can replace the lap time simulation platform as a fast performance prediction tool. Using such a fast prediction tool, searching algorithms can be utilized to develop a FE race strategy. So, the ANN prediction model works as an evaluation function in the searching algorithms. In the next section, Monte Carlo tree search method is introduced as an intelligent search technique to be used for FE racing strategy development in combination with ANN.

4 Race planning and optimization using Mont Carlo tree search

4.1 Monte Carlo tree search (MCTS) method

Monte Carlo tree search is a searching algorithm based on Markov decision processes (MDP) [48] and Monte Carlo method trying to find optimal decisions in a given space. While intermediate states do not need to be evaluated in the Monte Carlo simulation process, MCTS is very good at shrinking large-sized problems by only taking the reward from the terminal state at the end of simulation process, then balancing the exploitation and exploration. Therefore, MCTS is very suitable for solving the previously raised energy management strategy problem. Although the proposed ANN saves a huge amount of computation time in performance predictions, still it is impossible to use an exhaustive 'direct search' for such a big-scale strategy problem. This is the reason why an advanced search technique like MCTS is used here.

The MCTS is completed by running a number of MCTS iterations. As shown in Fig. 9, each MCTS iteration comprises four processes, selection, expansion, simulation and backpropagation. Each time the tree agent searches into a deeper layer, it is considered as a lap finished in the race. In this study, a 32-lap race means the tree has a maximum depth of 32 layers.

Starting from the root, the agent selects among the parent's child nodes and then moves to the child node with the highest upper confidence bounds for trees (UCT) value [49]. It then takes that child as a new parent to proceed the selection process. The selection is terminated when the agent reaches unvisited child (leaf) node or a terminal state, which technically is also a leaf node. The UCT value in this study is calculated using the following equation:

$$\mathrm{UCT} = \frac{Q(s,a) - \min_{a\in A} Q(s,a)}{\max_{a\in A} Q(s,a) - \min_{a\in A} Q(s,a)} + C_p\sqrt{\frac{2\ln N(s)}{N(s,a)}} \tag{5}$$

where the reward $Q(s,a)$ is normalized as the exploitation term and parameter C_p is the balancing factor which in this study is assigned $\sqrt{2}$ as default according to Kocsis and Szepesvári [49].

After selection, if the agent ends up at a node which has been visited but does not have any child yet, the node will be expanded according to the tree policy. In this section, tree policy is a full expansion policy which expands all the possible actions in the action space based on the current state (i.e., battery temperature, battery SOC and number of laps in our problem). Table 1 shows the defined possible choices of the four parameters that driver can change during a race.

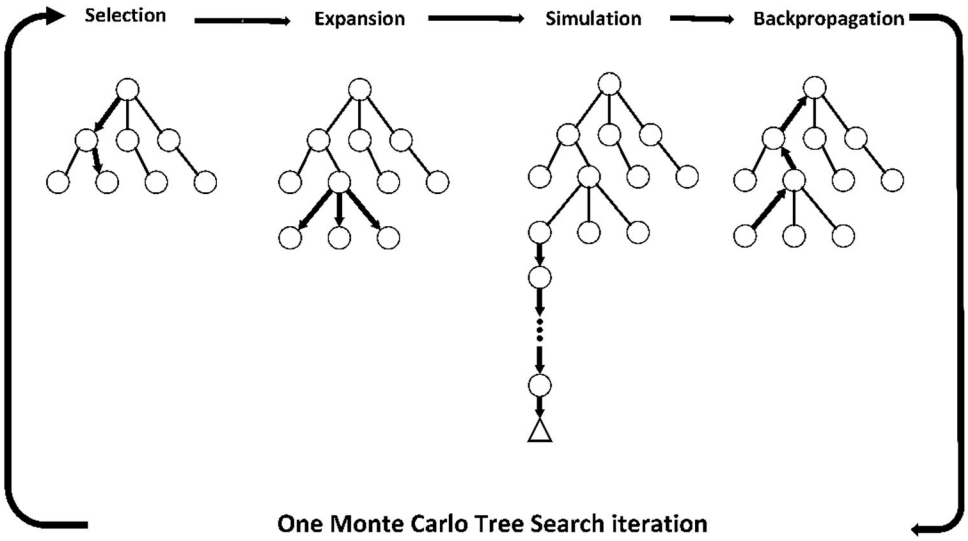

Fig. 9 Monte Carlo tree search iteration

Table 4 FE driver's choices of setting parameters

Parameter	Choices
Driver power (kW)	190, 195, 200, 225
Regeneration power (kW)	0, 100, 250
Coasting torque (Nm)	0, 100, 200
Coasting distance (m)	0, 60, 120

If a parent state has no constraints, its action spaces will contain 68 different setting combinations; thus, 68 child nodes will be created following the parent node. If a parent state has constraints such as battery temperature higher than 50 °C or 'Attack mode' already activated, the action space will be smaller resulting in less child nodes. While the child nodes are created, the child states will also be updated through the previously developed three prediction networks based on each child's setting configuration. By the end of the expansion process, the agent will move to the first expanded child node, which, according to Table 4, represents the choice of the lowest values for the four setting parameters.

After the expansion if the agent is at a node which has not been visited since being created, instead of expanding it, a simulation process will be completed starting from the node to the end of the tree. The simulation process starts from the state of the node. Three prediction networks will iterate for the remaining laps of race. In each iteration, the setting configuration is randomly picked from the action space based on the starting state of that iteration.

By the end of the simulation process, the SOC, battery temperature (T_{Bat} in °C) and race finishing time (t in seconds) results are used to calculate the reward through the reward function in the following:

$$R(s,a) = \begin{cases} 2800 - t, & \text{SOC} > 0 \text{ and } T_{\text{Bat}} < 60 \\ 0, & \text{else} \end{cases} \quad (6)$$

The reward will be greater when the car uses shorter time to successfully finish the race, while a DNF results in reward of 0.

In the backpropagation process, the number of times a node has been visited is firstly updated. The parameter $N(s,a)$ of the newly simulated node is assumed to be 1 instead of default 0 when created; meanwhile, the value of $N(s,a)$ of all its parent nodes will increase by 1 due to the new node. After the visiting times are updated, the UCT value of each node is upgraded based on the new Q-value and new visiting times.

4.2 Application of MCTS in different race scenarios

The complete MCTS is coded and tested to see how it reacts to different possible scenarios both before and during a race. The following result is obtained after 100,000 MCTS iterations.

4.2.1 Pre-race planning

A problem is raised in Sect. 2.1 that if no strategy was planned and driver kept pushing throughout the race, a DNF would be the result due to an over-heated flat battery. As a pre-race application, the MCTS will search for entire 32 layers of search depths. The MCTS solution is shown in figures.

It can be seen from Fig. 10 that MCTS instructs that 'Attack Mode' needs to be activated at lap 3 and later lower driver power will be needed at certain laps, and driver needs to do coasting during the race. Meanwhile, MCTS also gives the recommended regeneration power and coasting torque. The race result shown in Fig. 11 demonstrates that the MCTS solution has successfully slowed down the battery temperature rise and the energy consumption. The race was successfully finished with SOC of 0.88% and battery temperature of 59.8 °C at the end of the race.

A pre-race plan has been developed using MCTS. However, uncertainties will lead to some unexpected scenarios during the race. The following sections demonstrate the capability of MCTS solving different problems that might happen during a race.

4.2.2 Aggressive driving

This scenario can be triggered by an aggressive driver trying to chase one of his rivals. It is assumed that driver keeps pushing after the 'Attack Mode' laps at highest drive power with no coasting for 6 more laps (the 'A' area shown in Fig. 12) and comes back to the pre-race plan after those laps. The result of this aggressive driving and the MCTS reaction is shown in Fig. 13. The dashed lines in Fig. 13 show that the aggressive driving has resulted in a flat and over-heated battery which led to a DNF with the pre-race plan. After the aggressive laps, MCTS gave a solution that driver needs to lower down the drive power and do more coasting in the remaining laps (the 'B' area shown in Fig. 12). Despite that the race finishing time will be longer than what was planned, the MCTS has been able to eliminate the DNF crisis.

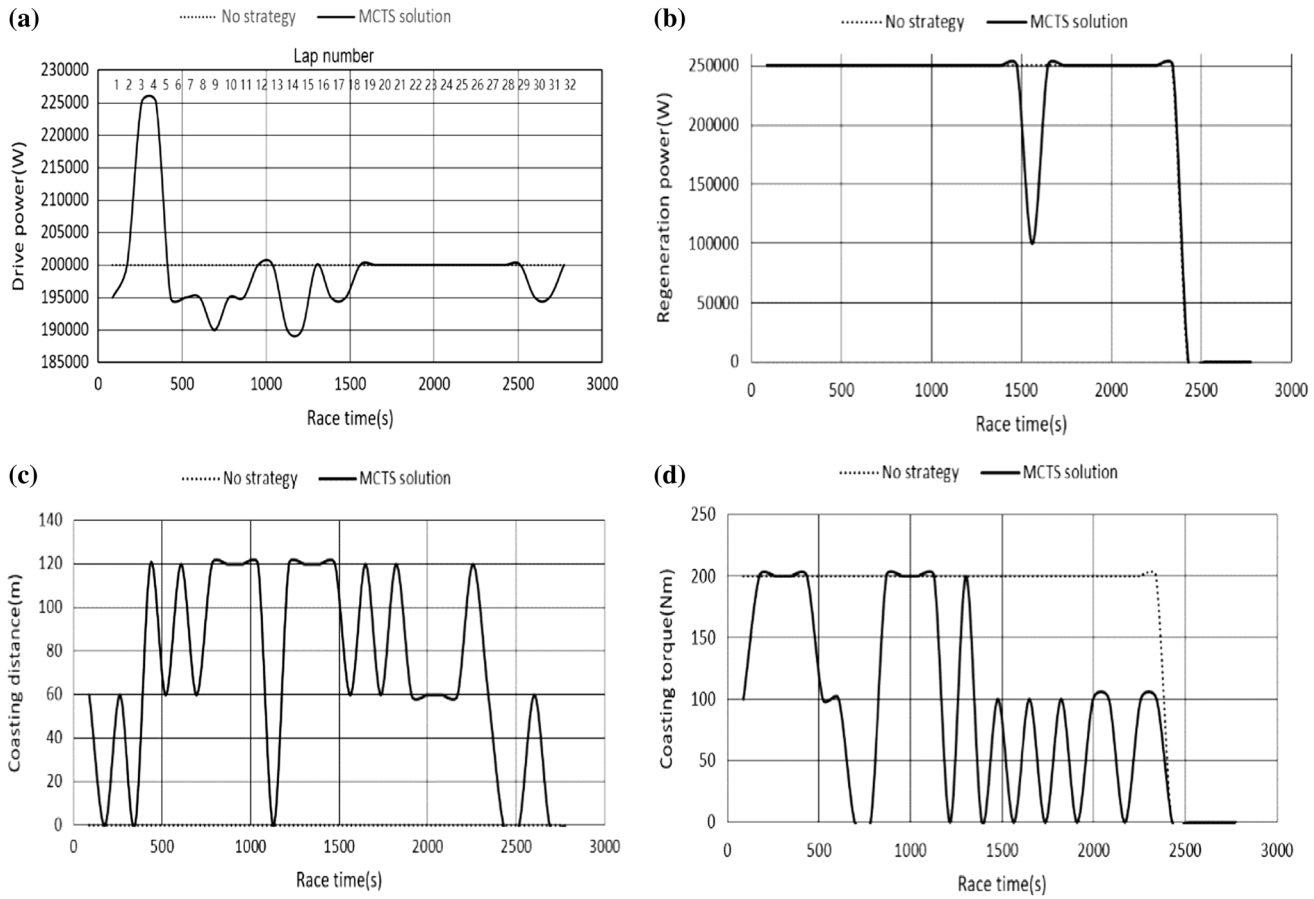

Fig. 10 MCTS solution, **a** drive power, **b** regeneration power, **c** coasting distance, **d** coasting torque

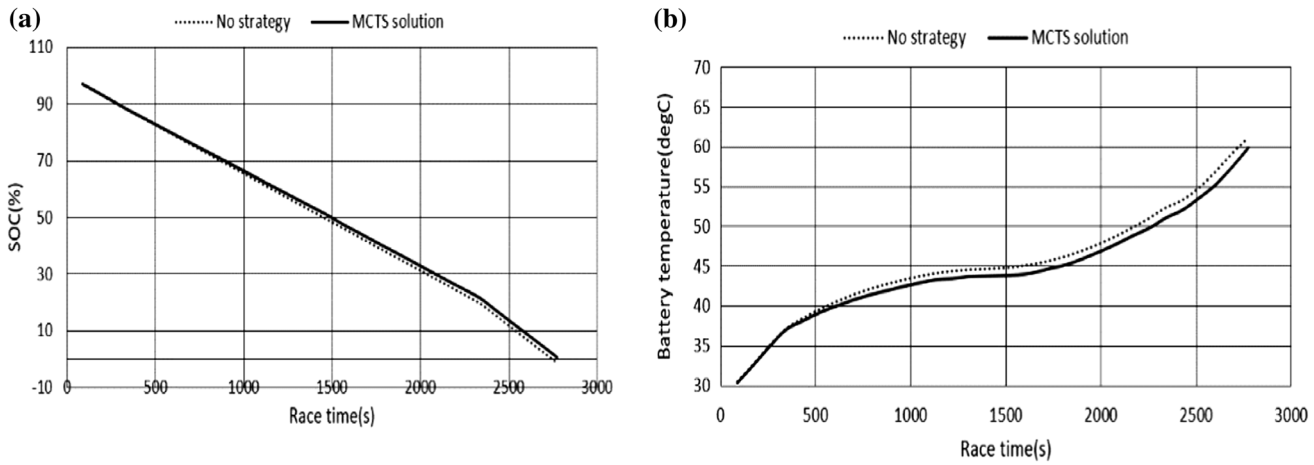

Fig. 11 Race results comparison: **a** battery SOC, **b** battery temperature

4.2.3 Different 'Attack mode' lap

It is probable that a heavy traffic will reduce the worthiness of activating 'Attack mode' and later cleaner laps will more benefit the 'Attack mode' in terms of lap time. It is assumed that driver decides to activate 'Attack mode' in lap 16–17 (the 'A' area shown in Fig. 14a) instead of pre-race planned lap 3–4. According to the prediction model, this resulted in a different battery temperature trend (the 'A' area shown in Fig. 15a) and an over-heated battery before the race finished (dashed line in Fig. 15b). After the 'Attack mode' laps, MCTS gives a solution in which slight

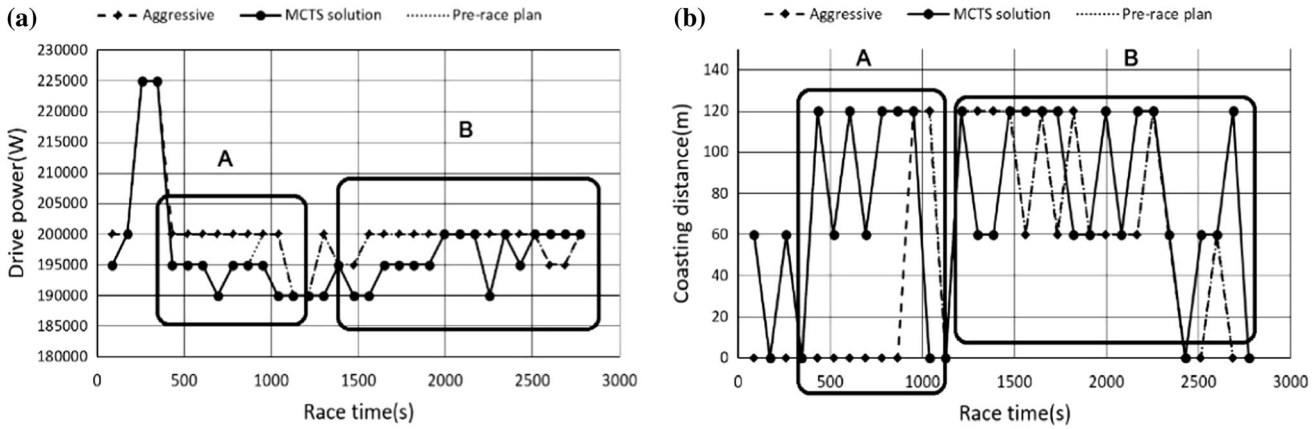

Fig. 12 Racing commands: **a** drive power, **b** coasting distance

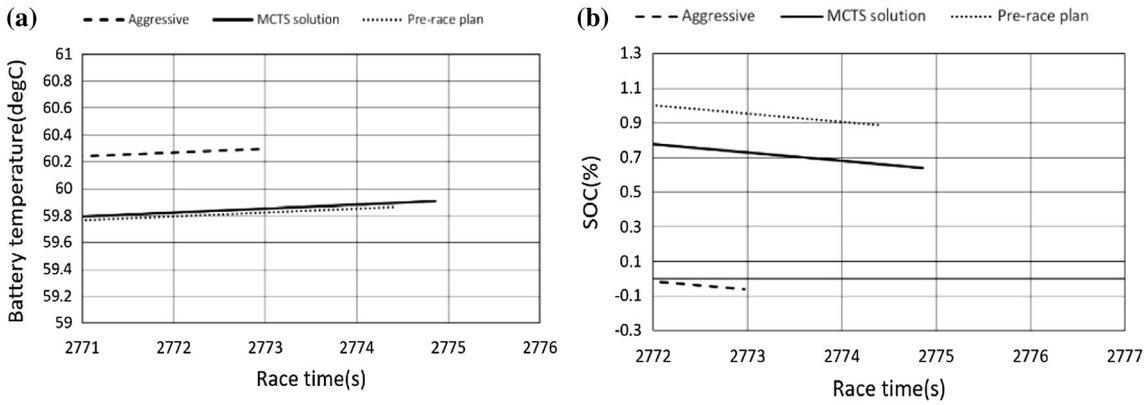

Fig. 13 Race results: **a** battery temperature, **b** battery SOC

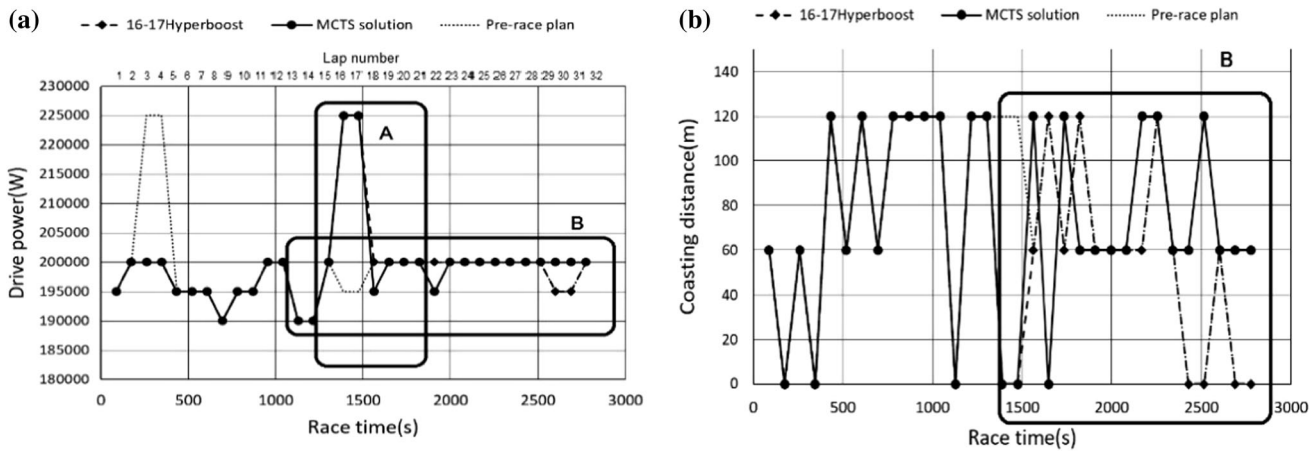

Fig. 14 Race commands: **a** drive power, **b** coasting distance

lower drive power and more coasting are required. In this way, the battery temperature was kept below 60 °C till the end of the race. However, it can be observed in Fig. 15b that the MCTS solution has found a faster way to finish the race which suggests that the pre-race plan was not the optimal solution. This will be discussed later.

4.2.4 Safety car scenario

In modern motor racing, especially the narrow street-based FE races, it is very likely that incidents would happen which result in safety cars being deployed. When the safety car is out, drivers will be driving in a very slow pace on the

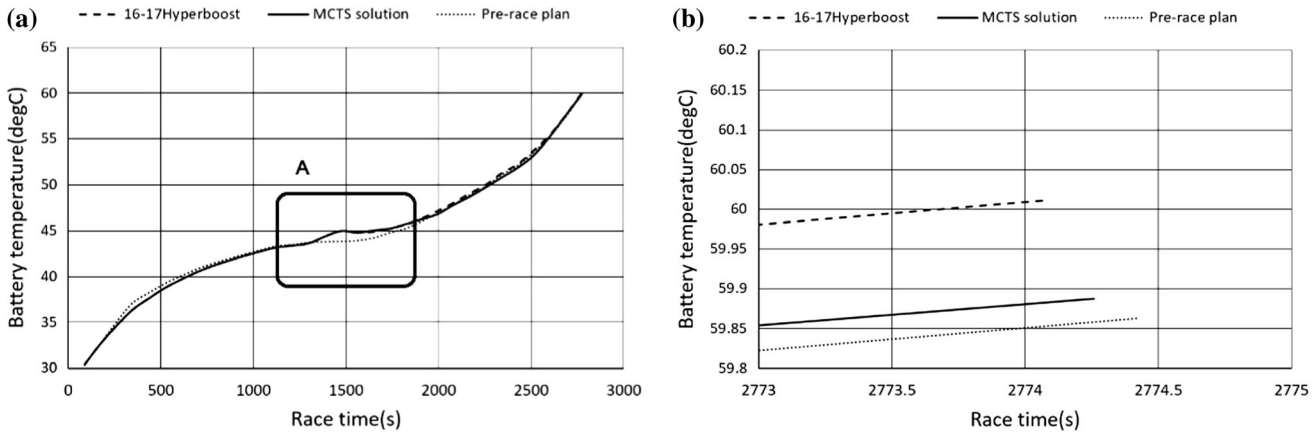

Fig. 15 Race results: **a** battery temperature, **b** temperature at the end of race

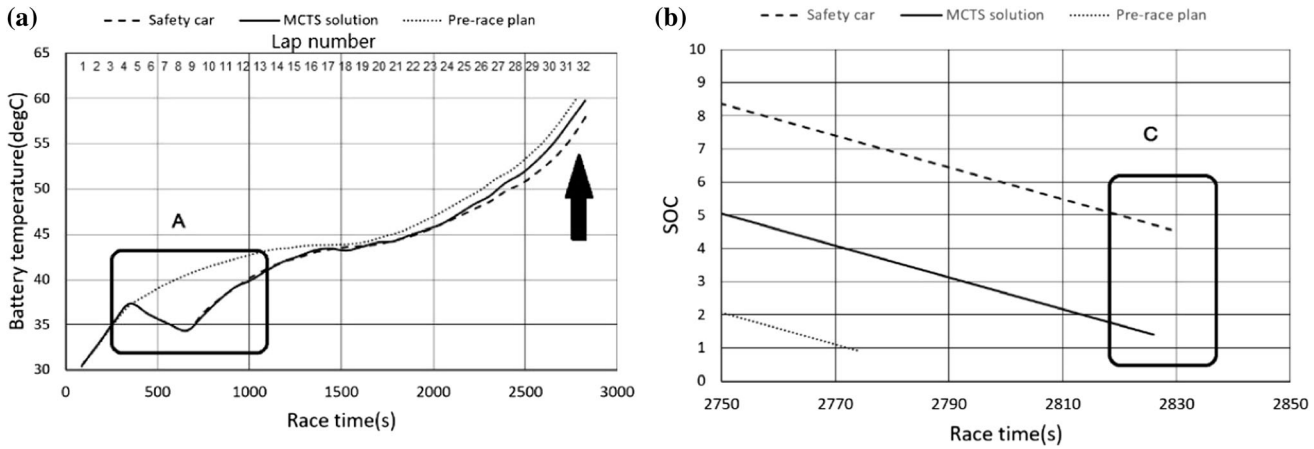

Fig. 16 Race results: **a** battery temperature, **b** battery SOC

track meaning that the energy consumption and heat generation will be very low during those laps. It can be seen from the 'A' area shown in Fig. 16a that during lap 5–7 when safety car is out, the battery temperature is cooled down. If driver keeps driving in the planned way, by the end of the race, the battery temperature will have a big margin from over-heating and there will be plenty of energy left as shown in the 'C' area shown in Fig. 16b. After the safety car went in, the MCTS gave a solution as shown in the 'A' area in Fig. 17a and the 'B' area in Fig. 17b. As a result, there will be no more need for compromise in drive power output and also there will be less need for regeneration. The new MCTS solution after safety car laps proved to be 3 s faster (the 'C' area shown in Fig. 16b) comparing to the original pre-race plan.

4.2.5 Environment change

Weather is one of the most unpredictable and uncontrollable factors during a race. The ambient temperature has a direct impact on the cooling system. Therefore, if the ambient temperature changes, reaction has to be taken to adapt to the new environment. Figure 18a shows a scenario where the ambient temperature rises by 2 °C at lap 10. It can be observed that the battery temperature rises faster than what was predicted before the race. If the driver keeps driving in the planned way, an over-heated battery will cause a DNF before end of the race. In order to eliminate the crisis, the MCTS has generated a solution as shown in Fig. 19. To keep the battery temperature in a safe range, drive power needs to be lowered down and a lot more coasting is required (the 'B' area shown in Fig. 19). This resulted in a 2-s slower race time (shown by arrow B in Fig. 18b) but no more DNF threats.

4.3 Assumptions and discussion

From the previous sections, it is clearly observed that such integration of ANNs and MCTS successfully generated solutions to both pre-race and in-race scenarios. In the pre-race planning scenario, the MCTS generated solutions for all four strategy-defining parameters (driving power,

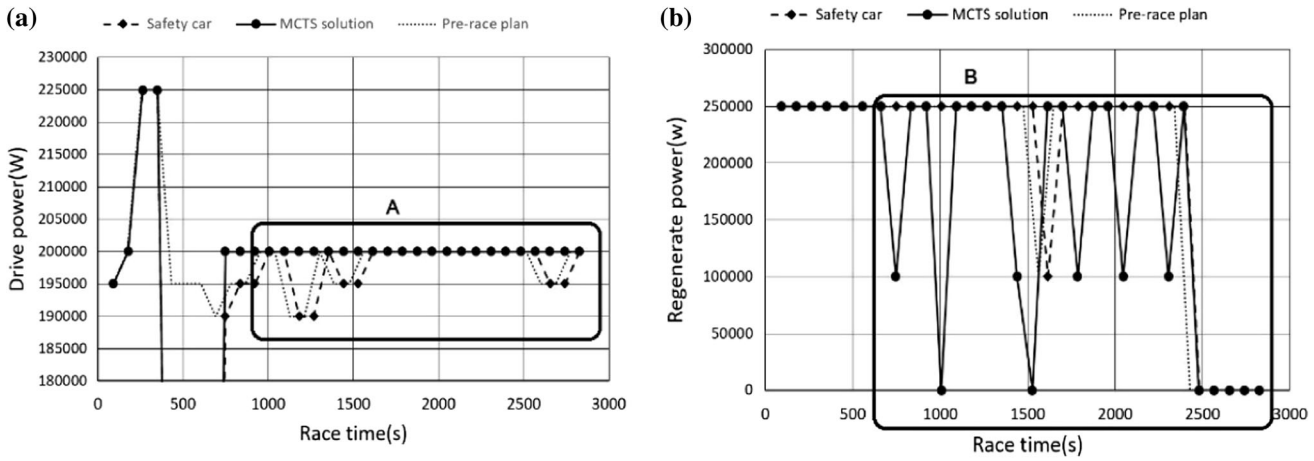

Fig. 17 Race commands: **a** drive power, **b** regeneration power

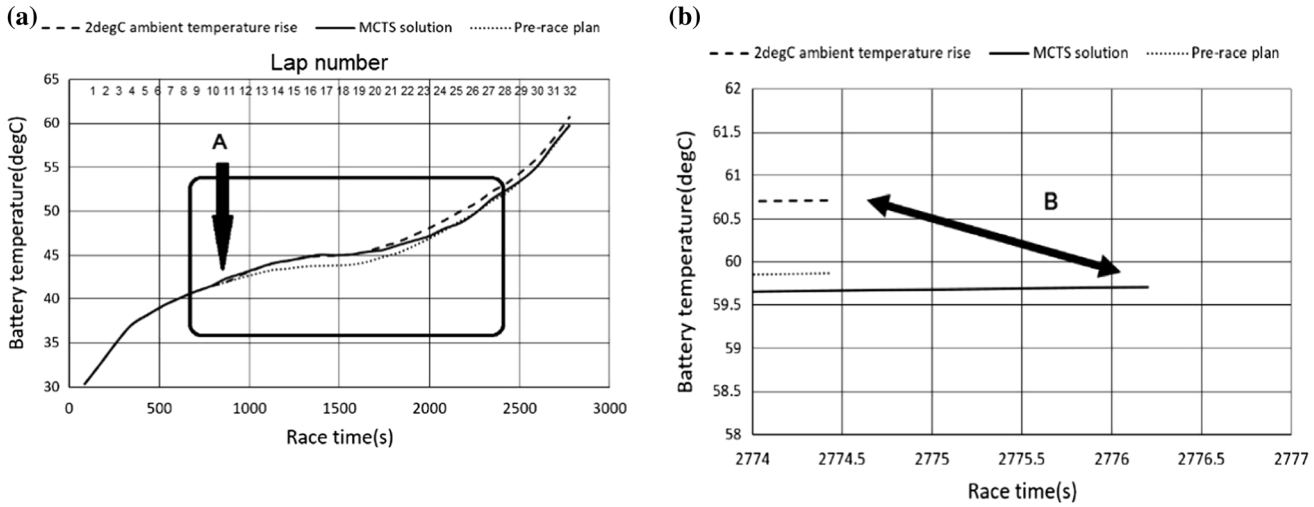

Fig. 18 Race results: **a** battery temperature, **b** temperature at the end of race

regeneration power, coasting distance and coasting torque). From the SOC and battery temperature, it can be observed that the MCTS utilized the full working range of the battery to generate a fast race finishing strategy which gives a reference for the following in-race scenarios.

To create an aggressive driving problem, it was assumed that the driver kept pushing after the 'Attack mode' for a number of laps. The reason for such assumption is that in real world it is very common that a driver failed to overtake his opponent with the help of 'Attack mode' but could get closer within those two laps. So it is rational to make such assumptions to mimic the attempt of a real driver keep trying to overtake his opponent. The number of aggressive laps was set to 6 to create a significant impact to the pre-race plan as can be observed from the figures that such changing would lead to a DNF. More laps of aggressive driving would not be rational in this case because in real world it would be very likely to raise other issues such as brake overheating or tire degradation and such irrational driver would not be realistic.

The case of different 'Attack mode' activation represents another common scenario in-race event that the best overtake opportunity not always appears as planned. Usually drivers have to drive normally to wait for that opportunity to raise. Therefore, to mimic such problem, the 'Attack mode' was set to be activated later in the race. One feature of interest is observed in Fig. 15b that when MCTS tried to eliminate the DNF threat, it found a faster solution to finish the race which means the pre-race plan was not the optimal. The reason behind can be explained as such that it is naturally very hard for MCTS to generate the optimal solution given such a big scale of problem. This shows the potential that such algorithm can be improved. Different ways have been proposed in relevant studies. A part of future work can focus on such improvement on the race strategy application.

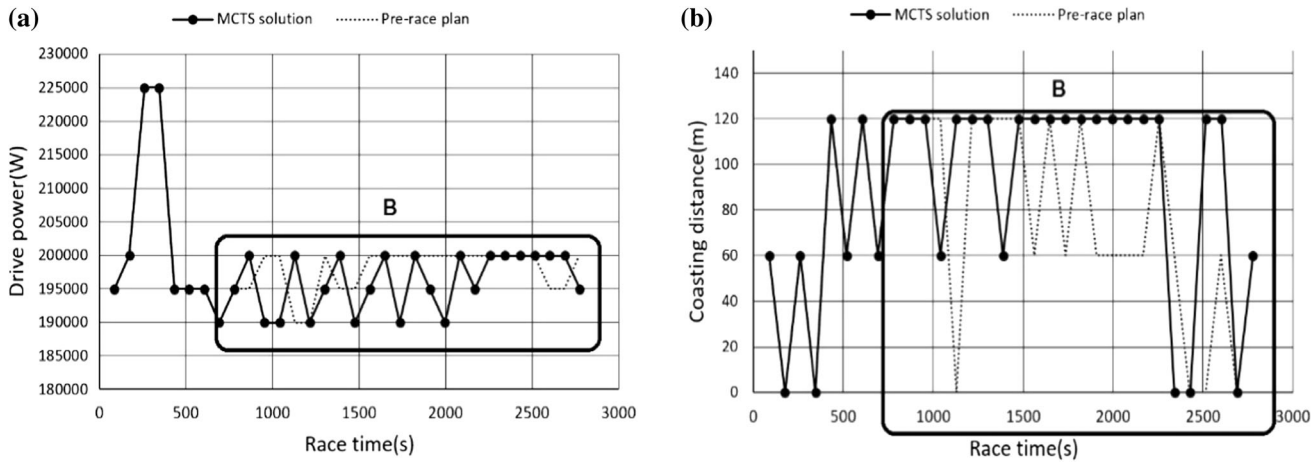

Fig. 19 Race commands: **a** drive power, **b** coasting distance

Incidents like crashes are very likely to happen in the street circuit racings especially in the early stages when drivers are closed and fights to overtake each other. Therefore, in the third in-race scenario, it was assumed an incident happened and safety car was deployed for 2 laps to wait for any hazards to be eliminated before the race returns to normal. Under the safety car period, the energy consumption and battery temperature rise are much lower than normal, which leaves a big margin from DNF to work on. From the result it can be seen that the MCTS successfully took advantage of that margin and updated the strategy to allow the driver to finish the race 3 s faster, which is a big improvement in motorsport application, compared to sticking to the pre-race plan.

The final scenario represents another tricky issue in the real world when race events are held in a place where weather is significantly changeable. The weather change would very likely to change the finish of the race because this basically changed the conditions for thermal management of the battery. The creation of such scenario in this study aimed to test the algorithm's capability of reacting to such weather changes and the MCTS succeeded. However, in real world this could be more complicated because the weather would also change the track condition which also has a big impact on the vehicle performance. This has the potential to become a part of future work.

5 Conclusions and future work

In this study, artificial neural network prediction models were built to predict the FE car performance. The effect of ANN depth and number of outputs on prediction accuracy was studied. Deeper ANNs produce less prediction errors, and separating the lap time, battery temperature and battery SOC prediction ANNs contributes to higher accuracy than using a single ANN to predict three parameters simultaneously. In terms of computational time, the ANN prediction models can produce more than 300,000 results within 20 s, while a traditional lap time simulation software can only produce one. The ANNs proved to be a powerful replacement tool of not only a single-lap simulation but also a race (multiple consecutive laps) simulation with decent accuracy and huge advantage in terms of computational speed. This has made it possible to run an advanced searching algorithm based on a big action space to solve the multi-stage strategy problem.

A Monte Carlo tree search algorithm was coded integrating the ANN prediction models as reward function. The MCTS is able to generate decent strategic solutions from the start of a race as pre-race planning. Additionally, it can also make quick and high-quality reactions to unexpected scenarios in a race, such as improper driving, safety car and ambient condition changes, eliminating potential battery temperature or energy crisis in order to finish a race successfully and competitively.

Based on the current study, there are limitations which could be improved in future work in such directions. First, the strategy solution of this MCTS was not the optimal. Further work can focus on improving the tree policy, rollout policy and searching efficiency to improve the overall performance of the algorithm. Second, apart from the 7 inputs of the neural network, there are other parameters that can affect the vehicle performance such as track condition, relative race position, tire condition. These variables will further increase the scale of problem and computational cost. But success of bringing these into consideration would make this algorithm more powerful. Third, the algorithm architecture proposed in this paper gives the solution within seconds which is fast enough for pit-wall applications. However, considering the nature of motorsport, if this algorithm can be applied as a part of on-

board system can calculate solution in real time, this would more significantly benefit the strategy.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Formula E opens with spectacular crash involving Nick Heidfeld and Nicolas Prost as Lucas di Grassi claims win—Telegraph (2014). https://www.telegraph.co.uk/sport/motorsport/11094128/Formula-E-opens-with-spectacular-crash-involving-Nick-Heidfeld-and-Nicolas-Prost-as-Lucas-di-Grassi-claims-win.html. Accessed 28 July 2018
2. What Does the Cockpit of a Formula E Car Look Like?—The Drive (2017). Accessed 28 July 2018 http://www.thedrive.com/flat-six-society/14665/what-does-the-cockpit-of-a-formula-e-car-look-like
3. Mi C, Masrur MA, Gao DW (2011) Hybrid electric vehicles: principles and applications with practical perspectives. Wiley, New York. https://doi.org/10.1002/9781119998914
4. Zhang X, Mi C (2011) Vehicle power management. Power systems. Springer, New York
5. Zhang B, Mi CC, Zhang M (2011) Charge-depleting control strategies and fuel optimization of blended-mode plug-in hybrid electric vehicles. IEEE Trans Veh Technol 60(4):1516–1525. https://doi.org/10.1109/tvt.2011.2122313
6. Moura SJ, Fathy HK, Callaway DS, Stein JL (2011) A stochastic optimal control approach for power management in plug-in hybrid electric vehicles. IEEE Trans Control Syst Technol 19(3):545–555. https://doi.org/10.1109/tcst.2010.2043736
7. Fotouhi A, Yusof R, Rahmani R, Mekhilef S, Shateri N (2014) A review on the applications of driving data and traffic information for vehicles' energy conservation. Renew Sustain Energy Rev J 37:822–833. https://doi.org/10.1016/j.rser.2014.05.077
8. Montazeri M, Fotouhi A, Naderpour A (2012) Driving segment simulation for determination of the most effective driving features for HEV intelligent control. J Veh Syst Dyn 50(2):229–246. https://doi.org/10.1080/00423114.2011.577898
9. Chen Z, Mi CC, Xu J, Gong X, You C (2014) Energy management for a power-split plug-in hybrid electric vehicle based on dynamic programming and neural networks. IEEE Trans Veh Technol 63(4):1567–1580. https://doi.org/10.1109/tvt.2013.2287102
10. Chen Z, Mi CC (2009) An adaptive online energy management controller for power-split HEV based on dynamic programming and fuzzy logic. In: 5th IEEE vehicle power and propulsion conference, VPPC'09, pp 335–339. https://doi.org/10.1109/vppc.2009.5289831
11. Schouten NJ, Salman MA, Kheir NA (2002) Fuzzy logic control for parallel hybrid vehicles. IEEE Trans Control Syst Technol 10(3):460–468. https://doi.org/10.1109/87.998036
12. Borhan H, Vahidi A, Phillips AM, Kuang ML, Kolmanovsky IV, Di Cairano S (2012) MPC-based energy management of a power-split hybrid electric vehicle. IEEE Trans Control Syst Technol 20(3):593–603. https://doi.org/10.1109/tcst.2011.2134852
13. Gong Q, Li Y, Peng ZR (2009) Trip based optimal power management of plug-in hybrid electric vehicle with advanced traffic modeling. SAE Int J Engines 1(1):861–872. https://doi.org/10.4271/2008-01-1316
14. Murphey YL, Park J, Kiliaris L, Kuang ML, Masrur MA, Phillips AM et al (2013) Intelligent hybrid vehicle power control part II: online intelligent energy management. IEEE Trans Veh Technol 62(1):69–79. https://doi.org/10.1109/tvt.2012.2217362
15. Murphey YL, Kuang ML, Masrur MA, Phillips AM (2012) Intelligent hybrid vehicle power control—part i: machine learning of optimal vehicle power. IEEE Trans Veh Technol 61(8):3519–3530. https://doi.org/10.1109/tvt.2012.2206064
16. Kiliaris L, Kuang ML, Masrur MA, Phillips AM, Murphey YL (2009) Intelligent vehicle power control based on machine learning of optimal control parameters and prediction of road type and traffic congestion. IEEE Trans Veh Technol 58(9):4741–4756. https://doi.org/10.1109/tvt.2009.2027710
17. Montazeri-Gh M, Fotouhi A, Naderpour A (2011) Driving patterns clustering based on driving features analysis. Proc Inst Mecha Eng Part C J Mech Eng Sci 225(6):1301–1317
18. Huang X, Tan Y, He X (2011) An intelligent multifeature statistical approach for the discrimination of driving conditions of a hybrid electric vehicle. IEEE Trans Intell Transp Syst 12:453–465. https://doi.org/10.1109/tits.2010.2093129
19. Montazeri-Gh M, Fotouhi A (2011) Traffic condition recognition using k-means clustering method. Int J Sci Iran Part B 18(4):930–937
20. Qiuming G, Yaoyu L, Zhongren P (2009) Power management of plug-in hybrid electric vehicles using neural network based trip modeling. In: American control conference, pp 4601–4606. https://doi.org/10.1109/ACC.2009.5160623
21. Yuan X, Wang J, Colombage K (2012) Torque distribution strategy for a front and rear wheel driven electric vehicle. In: 6th IET international conference on power electronics, machines and drives (PEMD 2012), vol 2(8): pp C32. https://doi.org/10.1049/cp.2012.0316
22. Gu J, Ouyang M, Lu D, Li J, Lu L (2013) Energy efficiency optimization of electric vehicle driven by in-wheel motors. Int J Autom Technol 14(5):763–772. https://doi.org/10.1007/s12239-013-0084-1
23. Wang R, Chen Y, Feng D, Huang X, Wang J (2011) Development and performance characterization of an electric ground vehicle with independently actuated in-wheel motors. J Power Sources 196(8):3962–3971. https://doi.org/10.1016/j.jpowsour.2010.11.160
24. Li X, Chen Y, Wang J (20012) In-wheel motor electric ground vehicle energy management strategy for maximizing the travel distance. In: Proceedings of the American control conference
25. De Novellis L, Sorniotti A, Gruber P (2013) Optimal wheel torque distribution for a four-wheel-drive fully electric vehicle. SAE Int J Passeng Cars Mech Syst 6:128–136. https://doi.org/10.4271/2013-01-0673
26. Pennycott A, De Novellis L, Sabbatini A, Gruber P, Sorniotti A (2014) Reducing the motor power losses of a four-wheel drive,

fully electric vehicle via wheel torque allocation. Proc Inst Mech Eng Part D J Autom Eng 228(7):830–839. https://doi.org/10.1177/0954407013516106
27. Lin C, Xu Z (2015) Wheel torque distribution of four-wheel-drive electric vehicles based on multi-objective optimization. Energies 8(5):3815–3831. https://doi.org/10.3390/en8053815
28. McCulloch WS, Pitts WA (1943) logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5(4):115–133. https://doi.org/10.1007/bf02478259
29. Tesauro G (1992) Practical issues in temporal difference learning. Mach Learn 8(3):257–277. https://doi.org/10.1023/a:1022624705476
30. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 5(2):157–166. https://doi.org/10.1109/72.279181
31. Hinton GE, Osindero S, Teh YW (2006) Fast learning algorithm for deep belief nets. Neural Comput 18:1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527
32. LeCun YA, Bengio Y, Hinton GE (2015) Deep learning. Nature 521(7553):436–444. https://doi.org/10.1038/nature14539
33. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G et al (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529(7587):484–489. https://doi.org/10.1038/nature16961
34. Chaslot G et al (2008) Monte-Carlo tree search: a new framework for game AI. In: AIIDE
35. Browne C, Powley E (2012) A survey of monte carlo tree search methods. IEEE Trans Intell AI Games 4(1):1–49. https://doi.org/10.1109/tciaig.2012.2186810
36. Kocsis L, Szepesvári C (2006) Bandit based Monte-Carlo planning. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)
37. Fotouhi A, Auger DJ, Propp K, Longo S (2018) Lithium-sulfur battery state-of-charge observability analysis and estimation. IEEE Trans Power Electron 33(7):5847–5859. https://doi.org/10.1109/TPEL.2017.2740223
38. Jongerden M, Haverkort B (2008) Battery modeling. Technical report in Faculty Electrical Engineering, Mathematics and Computer Science. https://doi.org/10.1109/mc.2003.1250886
39. Shafiei A, Momeni A, Williamson SS (2011) Battery modeling approaches and management techniques for Plug-in hybrid electric vehicles. In: Vehicle power and propulsion conference (VPPC), 2011. IEEE. https://doi.org/10.1109/vppc.2011.6043191
40. Fotouhi A, Auger DJ, Propp K, Longo S, Wild M (2016) A review on electric vehicle battery modelling: from lithium-ion toward lithium–sulphur. Renew Sustain Energy Rev. https://doi.org/10.1016/j.rser.2015.12.009
41. Bernardi D (1985) A general energy balance for battery systems. J Electrochem Soc. https://doi.org/10.1149/1.2113792
42. Jaguemont J, Boulon L, Dubé Y (2016) A comprehensive review of lithium-ion batteries used in hybrid and electric vehicles at cold temperatures. Appl Energy. https://doi.org/10.1016/j.apenergy.2015.11.034
43. Vertiz G, Oyarbide M, Macicior H, Miguel O, Cantero I, De Arroiabe PF et al (2014) Thermal characterization of large size lithium-ion pouch cell based on 1d electro-thermal model. J Power Sources. https://doi.org/10.1016/j.jpowsour.2014.08.092
44. Liu H, Wei Z, He W, Zhao J (2017) Thermal issues about Li-ion batteries and recent progress in battery thermal management systems: a review. Energy Conver Manag. https://doi.org/10.1016/j.enconman.2017.08.016
45. Barnard E, Wessels LFA (1992) Extrapolation and interpolation in neural network classifiers. IEEE Control Syst. https://doi.org/10.1109/37.158898
46. Hagan MT, Menhaj MB (1994) Training feedforward networks with the marquardt algorithm. IEEE Trans Neural Netw. https://doi.org/10.1109/72.329697
47. Dan Foresee F, Hagan MT (1997) Gauss-Newton approximation to bayesian learning. In: IEEE international conference on neural networks—conference proceedings. https://doi.org/10.1109/icnn.1997.614194
48. Bellman R (1957) A Markovian decision process. J Math Mech. https://doi.org/10.1007/bf02935461
49. Kocsis L, Szepesvári C, Willemson J (2006) Improved Monte-Carlo search. AAAI/IAAI. https://doi.org/10.1007/11871842_29

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

https://dspace.org/

Formula-E race strategy development using artificial neural networks and Monte Carlo Tree Search

Liu, Xuze; Fotouhi, Abbas

2020-03-30
Springer
https://doi.org/10.1007/s00521-020-04871-1
Liu X, Fotouhi A. (2020) Formula-E race strategy development using artificial neural networks and Monte Carlo Tree Search. Neural Computing and Applications, Volume 32 Issue 18, September 2020, pp. 15191-15207

Downloaded from a DSpace repository. https://dspace.org/