Article

FPGA Implementation of an Ant Colony Optimization Based SVM Algorithm for State of Charge Estimation in Li-Ion Batteries †

Mattia Stighezza , Valentina Bianchi * and Ilaria De Munari

Department of Engineering and Architecture, University of Parma, 43121 Parma, Italy; mattia.stighezza@unipr.it (M.S.); ilaria.demunari@unipr.it (I.D.M.)
* Correspondence: valentina.bianchi@unipr.it; Tel.: +39-0521-906284

† This paper is an extended version of our paper published in: Stighezza, M.; Bianchi, V.; De Munari, I. HDL Code Generation from SIMULINK Environment for Li-Ion Cells State of Charge and Parameter Estimation, Lecture Notes in Electrical Engineering. In Proceedings of the 2021 Conference on Applications in Electronics Pervading Industry, Environment and Society, APPLEPIES 2020 Virtual, Online, 19–20 November 2020; Volume 738, pp. 136–143.

Abstract: Monitoring the State of Charge (SoC) in battery cells is necessary to avoid damage and to extend battery life. Support Vector Machine (SVM) algorithms and Machine Learning techniques in general can provide real-time SoC estimation without the need to design a cell model. In this work, an SVM was trained by applying an Ant Colony Optimization method. The obtained trained model was 10-fold cross-validated and then designed in Hardware Description Language to be run on FPGA devices, enabling the design of low-cost and compact hardware. Thanks to the choice of a linear SVM kernel, the implemented architecture resulted in low resource usage (about 1.4% of Xilinx Artix7 XC7A100TFPGAG324C FPGA), allowing multiple instances of the SVM SoC estimator model to monitor multiple battery cells or modules, if needed. The ability of the model to maintain its good performance was further verified when applied to a dataset acquired from different driving cycles to the cycle used in the training phase, achieving a Root Mean Square Error of about 1.4%.

Keywords: state-of-charge (SoC) estimation; battery management; FPGA; VHDL; ant colony optimization (ACO)

Citation: Stighezza, M.; Bianchi, V.; De Munari, I. FPGA Implementation of an Ant Colony Optimization Based SVM Algorithm for State of Charge Estimation in Li-Ion Batteries. *Energies* **2021**, *14*, 7064. https://doi.org/10.3390/en14217064

Academic Editors: Riccardo Berta, Sergio Saponara, Alessandro De Gloria and Francesco Bellotti

Received: 23 September 2021
Accepted: 25 October 2021
Published: 28 October 2021

1. Introduction

One of the most critical aspects of electric vehicles and modern automotive frameworks is battery management. In recent years, researchers' interest has been focused on new techniques that allow increases in battery performance and extend their lifetime [1,2].

Currently, battery cells are commonly based on Lithium-Ion and Lithium-Polymer chemistries [3]. These technologies are well established and allow both high energy and high power density. Hence, improving their performance and developing new chemical compositions is not a pressing task [3]. The study of new Battery Management Systems (BMS) [3–6] could lead to an improvement in the battery lifecycle through the monitoring of suitable battery cell indicators. Among these, the State of Charge (SoC) and the State of Health (SoH) are particularly important [7]. The estimation of SoC is of utmost importance in BMSs to make decisions about their management, and it can be an input for other forms of processing, such as SoH estimation or cell balancing [8]. It is worth noting that an advanced BMS should be able to monitor each cell of a battery pack to obtain an accurate estimation of every single SoC and SoH [9]. Since SoC and SoH cannot be directly measured and change continuously during the battery's life [8], suitable models have to be established, preferably operating in real-time.

The SoC can usually be evaluated according to two main approaches: model-based and data-driven [8,10,11].

Among model-based approaches, physical electrochemical models have been presented in research studies, but they are difficult to implement due to the huge amount of unspecified variables that need to be defined [11]. Moreover, a deep knowledge of the processes behind battery cell operation and its chemistry is required in order to construct a model that best fits the cell. The most common model-based approach, which guarantees higher accuracy and lower complexity, is the Equivalent Circuit Model (ECM) [12,13] where the battery cell is described as an electrical circuit with resistors and capacitors. In some implementations, SoC evaluation combines ECM and different methods, such as Kalman Filters (KF) or Proportional-Integral (PI) Observer [14], to improve accuracy despite the increase in computational demand [11]. Different implementations were presented in research based on microcontroller systems [15–17], but complex models and algorithms also led to solutions exploiting the intrinsic parallelism of FPGA devices [18–21]. For instance, in [21], the model-based design of a Mixed Algorithm (MA) combining ECM and Coulomb Counting (CC) for SoC estimation was proposed. The Simulink model was automatically converted into VHDL code through the HDL Coder Tool. The synthesized code resulted an area occupation of around 23%. Although the SoC was successfully evaluated with a Root Mean Square Error (RMSE) of about 1.5%, the result was strongly dependent on a correct battery parameters assessment, and it was subject to change, depending on the selected estimation procedure (e.g., Hybrid Pulse Power Characterization (HPPC) [22] and KF [23]). Typically, methods based on ECM and filtering allow maximum errors even below 1% [11], but obtaining suitable cell parameters for a model is a time-consuming process [8] that requires several experiments (e.g., to map the relationship of one parameter to another [8]). Moreover, these algorithms can require significant computational power [11]. In general, model-based techniques imply the identification of all the significant model parameters, and these approaches are difficult to implement in all types of batteries [8].

Instead, data-driven approaches mainly rely on Machine Learning (ML) algorithms working on empirical observations. Generated ML models provide relationships between on-field measurements of battery features (e.g., voltage, current, and temperature) and their SoC. In [24], different methods are presented with their specific advantages and disadvantages: adaptive methods, to which ML algorithms belong, generally provide higher accuracy but also require higher computational power for complex calculation [24]. Moreover, data-driven algorithms can prevent the use of battery models and added filters exploited in the model-based approaches, relying on self-learning parameters [8,24]. Furthermore, data-driven approaches allow greater flexibility: in fact, it is possible to train the algorithm with measurements from different kinds of batteries (e.g., different producers, models, and chemistries), and then apply the trained model to different cases (i.e., those considered in the training phase). On the other hand, data-driven algorithms usually require a huge amount of data. Some battery-related datasets have been provided in relevant studies [25–27], and they can help to overcome this drawback. Different ML algorithms have been employed for SoC estimation [28], such as Neural Networks (NN) [29–32] and Deep Learning (DL) [33,34]. Support Vector Machines (SVMs) have also been explored in the Support Vector Regression (SVR) version [35,36]. These algorithms have typically been tested on PC-based platforms [28,30,31,33,34,37]. In some works, implementations on microcontrollers are presented [32,35]. In [35], an RSME of about 2.5% was achieved. In general, in microcontroller implementations, a fixed data size can be exploited, limiting the model's flexibility. FPGA platforms offer some advantages, since performing real-time estimations through ML algorithms requires a large amount of computational power, which an FPGA can easily provide [38,39]. Moreover, the hardware flexibility and reconfigurability provided by FPGAs allow the implementation of multiple instances of the same algorithm in parallel. In fact, in the case of a battery pack, each cell module is usually monitored by a slave board belonging to a master BMS board [40]. Exploiting FPGA, slave boards could be implemented in the same FPGA platform equipped with multiple monitoring algorithms acting in parallel on each cell, reducing the hardware employed.

In the relevant research studies, there are examples of SVR [41,42] implementations on FPGA in many applications, but, to the best of our knowledge, not in the SoC estimation field. In [43], an NN model for SoC estimation was implemented on FPGA. However, the model resulted in high area occupancy (i.e., 40% of block arithmetic/DSP48E1 were exploited). Moreover, the need for DPS blocks limits the possibility of exporting the model on simple and low-cost FPGAs.

The main goal of our work is to implement a data-driven approach for SoC estimation on an FPGA platform, achieving comparable accuracy to PC-based implementations and improving performance in comparison with other ECM-embedded solutions, limiting the area occupation to allow, in the future, the parallelization of multiple instances of the developed algorithm on a single FPGA. Therefore, an SVR algorithm was selected and optimized through an Ant Colony Optimization (ACO) approach [44]. Other studies applied ACO optimization to SVR [45–47], but the SoC assessment field was not considered. The SVR obtained model was implemented on a Xilinx Artix 7 [48] XC7A100T-1CSG324C FPGA. Resources utilization was analyzed and timing simulations are reported. An RMSE of the proposed solution was evaluated on the implemented circuit through datasets including measured data.

The paper is organized as follows. In Section 2, the designed architecture is introduced, while in Section 3 the simulation results are discussed. Then, in Section 4, conclusions are drawn.

2. Materials and Methods

The SVMs were originally developed by V. Vapnik in 1995 for binary classification [49]. However, they can also be applied to regression problems (SVR) [50,51]. Given a set of N data samples $\boldsymbol{x}_i \in \mathrm{R}^{\mathrm{n}}$, and their corresponding known outputs $\boldsymbol{y}_i \in \mathrm{R}$, it is possible to construct a regression function by solving the SVR quadratic programming problem [51]. The resulting SVR estimation function for a new input vector x is

$$f(x) = \sum_{i=1}^{N_{SV}} (\alpha_i - \alpha_i{}^*)\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x_i}) + b \tag{1}$$

where α_i and $\alpha_i{}^*$ are the Lagrange multipliers. The N_{SV} samples associated with nonzero Lagrange multipliers are called Support Vectors (SV). $K(\boldsymbol{x},\boldsymbol{x}_i)$ is a kernel function that maps the input space R^{n} to a so-called high dimensional feature space R^{nk}, where linear regression is performed, and b is a bias term.

Some possible choices [52] for kernel functions in SoC estimation are:

- Linear: $K(\boldsymbol{x},\boldsymbol{x}_i) = \boldsymbol{x} \cdot \boldsymbol{x}_i$;
- Polynomial: $K(\boldsymbol{x},\boldsymbol{x}_i) = ((\boldsymbol{x} \cdot \boldsymbol{x}_i) + p)^d$;
- Gaussian Radial Basis Function (RBF): $K(\boldsymbol{x},\boldsymbol{x}_i) = exp(-\|\boldsymbol{x} - \boldsymbol{x}_i\|^2/2\sigma^2)$.

In this study, the MATLAB design suite was used for the training of the SVR model. A large collection of data acquired on a Panasonic 18650 Li-Ion battery cell, which is publicly available [25], was used. In [25], the current profiles and the related measurements were collected from real batteries applying some of the most popular drive cycles. In this study, the "Neural Network (NN) driving cycle" was used because it was specifically designed to be used with ML training processes [25].

The design of the SVR model was performed using the MATLAB *fitrsvm* function [53] and the set of training data mentioned above. Different features and kernels were tested during the training process to select the best compromise between high accuracy (i.e., low RMSE) and low implementation complexity. To test the possibility of also applying the trained model to datasets not included in the training, the different kernel solutions were validated on a dataset based on the US06 [54], one of the most aggressive highway driving cycles [55].

Therefore, a linear kernel was selected. The trained SVR linear kernel results in a model that can be used for successive inferences, starting from new input vectors, using the following estimation function:

$$f(x) = \alpha^T * \left(\frac{x}{KS} * \frac{SV^T}{KS}\right)^T + b \quad (2)$$

where KS is the Kernel Scale, a positive scalar by which the software divides all input data before applying kernel computation, α is the vector of stored values representing the differences between the two Lagrange multipliers for each SV, b is the scalar parameter in (1), and T is the transpose operator.

Next, to optimize the SVM model and achieve higher accuracy in SoC estimation, an Ant Colony Optimization (ACO) iterative algorithm was applied to tune the model parameters. This algorithm takes inspiration from ant foraging behavior [44]. When an ant finds food, i.e., a set of parameters of SVM, it deposits a given quantity of pheromones on the path, depending on the quality of food source, i.e., the quality of the model with the associated parameters (e.g., lower RMSE, higher quality). Other ants in the colony then move towards the point with the most quantity of pheromones, searching for a set of parameters near to the best one found up to that point. The process ends when a selected number of iterations is reached or the changes to the parameters are smaller than a given threshold.

The kernel scale KS, the approximation accuracy ε, and the box constraint C were selected for inclusion in the optimization process of the SVR model, since they are the same parameters considered by the built-in MATLAB optimization algorithms [53]. This offers a benchmark with which to evaluate the effectiveness of ACO. A detailed mathematical explanation of ε and C is beyond the scope of this paper, and it can be found in [50].

Once the ACO-optimized model was trained, a 10-fold cross validation was performed to evaluate the accuracy in terms of the maximum achieved error between the estimated and the expected SoC. For this purpose, the NN driving-cycle-based dataset was used for both the training and the test phases.

The model Equation (2) can be greatly simplified, considering the parameter β computed as:

$$\beta = \alpha^T * \frac{SV}{KS} \quad (3)$$

Furthermore, it is possible to compute a β_{scaled} parameter from β as

$$\beta_{scaled} = \frac{\beta}{KS} \quad (4)$$

Applying β and β_{scaled}, it is possible to transform Equation (2) in

$$f(x) = \langle \beta_{scaled}, x \rangle + b \quad (5)$$

Hence, the necessary hardware architecture to be implemented on an FPGA device to elaborate new input samples and perform a new SoC estimation results in a scalar product and sum of the b parameter, avoiding divider use, which is highly complex when implemented in hardware on an FPGA.

The hardware-implemented, ACO-optimized linear SVM model was finally evaluated against the US06-based dataset.

In Figure 1, an overview of the architecture of the proposed system is shown. For testing purposes, along with the SVR estimation model, a UART interface was implemented on the FPGA board. This communication port was conceived to feed the system with an input vector x at each time step. Once the proposed model has estimated the current SoC sample, the result is sent back to the PC to evaluate the corresponding error.

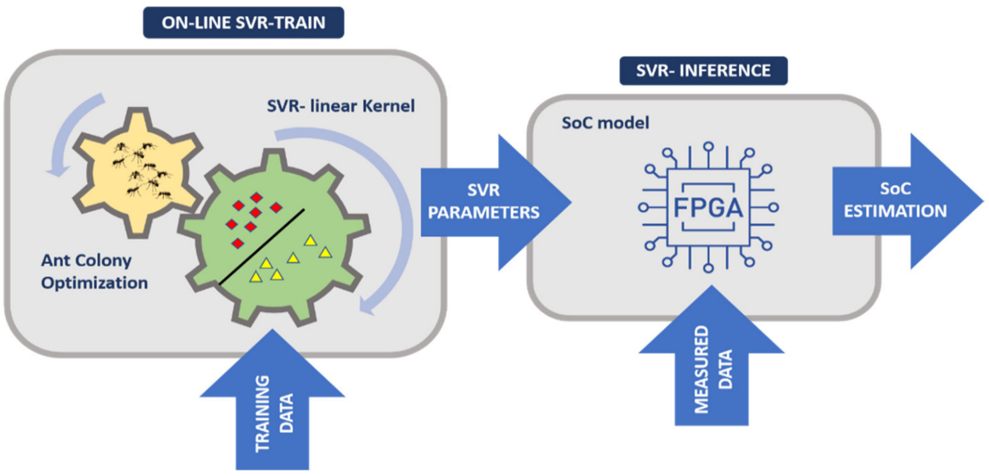

Figure 1. System architecture overview.

In Table 1, the datasets exploited at each training/test step are summarized.

Table 1. Summary of datasets used in each evaluation step, for training and test.

Phase	Train Set	Test Set
Evaluation of different Kernel functions	NN drive cycle	US06 drive cycle
Comparison between different MATLAB optimization strategies with ACO	NN drive cycle	US06 drive cycle
ACO-optimized Linear SVM cross-fold validation evaluation	NN drive cycle	NN drive cycle
FPGA-implemented ACO-optimized Linear SVM evaluation	Not performed	US06 drive cycle

3. Results and Discussion

The SVR model was trained by using the *fitrsvm* function and the "NN drive cycle" train dataset [25]. Simulations on MATLAB resulted in a better RMSE when the following four features were adopted:

- Current flowing through the cell (I)
- Cell voltage (V)
- Cell temperature (T)
- Estimation of SoC at the previous time step (prevSOC) [56]

The four selected features were used to evaluate the best kernel candidate for the system. Linear, Quadratic, Cubic, and Radial Basis Function (RBF) kernels were tested. For test purposes, the US06 drive cycle data [54] were exploited. The calculated RMSE, reported in Table 2, demonstrates that the best results were obtained with a linear kernel, which is also the kernel with the expected lower resource demand [57,58].

Table 2. RMSE results for four main types of SVM kernels on the "NN drive cycle" training dataset tested on the US06 drive cycle data.

Kernel	RMSE (%)
Linear	8.7
Quadratic	16.7
Cubic	35.9
Sigmoid	30.2
RBF	35.5

Once the linear kernel model was selected, the ACO algorithm was applied for the model optimization. In this study, the ant colony population M was set to 30 and each ant performed 30 moves (i.e., iterations) during the search process, starting from random positions. This choice resulted in the best compromise between model accuracy and processing time [59]. Each ant performed the *fitrsvm* function for the specified number of iterations, leading to a model with a specific combination of the KS, ε, and C parameters described above. The selected model was the one with the lowest RMSE. To assess the performance of the developed algorithm, it was compared with three built-in MATLAB optimization algorithms for SVR: "Bayesian", "Grid Search" and "Random Search". It is worth noting that the optimization result can change over different trials [53]. Therefore, each algorithm was run fifteen times and the mean RMSE value was evaluated on the US06 dataset. Results are shown in Table 3.

Table 3. RMSE and MAE results for a set of 15 simulations of different optimization algorithms.

Algorithm	RMSE (%)			MAE (%)		
	Min	**Mean**	**Max**	**Min**	**Mean**	**Max**
Bayesian	1.9	9.6	25.3	1.6	8.4	21.8
Grid Search	1.6	7.2	18.8	1.3	6.1	15.0
Random Search	1.4	6.8	53.6	1.2	5.9	46.5
ACO	1.4	3.9	7.2	1.2	3.2	5.8

The lowest RMSE was obtained with *"Random Search"* and ACO. Nevertheless, the mean RMSE for the first three algorithms was found to be worse than with the ACO. The same considerations can be applied to the MAE.

Next, the best ACO-optimized SVR model architecture was 10-fold cross-validated and the results were compared with a recent PC-based implementation of an SVM algorithm with Particle Swarm Optimization (PSO) [36]. K-fold cross-validation is a common approach to evaluating the effectiveness of a trained ML model in other studies [36]. In [36], a maximum error of 2.5% was obtained. The proposed approach offers a good performance, with a maximum error of 1.2%.

Hence, the trained model was translated into VHDL code to be programmed into a Xilinx Artix 7 [48] XC7A100T-1CSG324C FPGA. Equation (5) was coded in VHDL using a DSP-free architecture. This allows the evaluation of system feasibility on different types of FPGA boards, even on those that are not equipped with DSP slices, allowing the user to also choose cheaper boards if the resource usage is suitable. All the operations are performed with fixed-point precision, which can lead to a simpler hardware architecture. Signed 32-bit word length data with 23 fractional bits were chosen. This format can represent data in a range from -256 to 255.999 and a precision of 1.2×10^{-7}: these values are compliant with the most common battery cells and modules [60,61]. It is worth noting that this data format is suitable for modules with up to about 60 cells in series and it can also correctly represent SVM model parameters.

To validate the implemented module, the US06 test dataset was used. Using, in the validation phase, a dataset different from that used in the training phase also allows evaluation of the generalization capability of the approach.

A first behavioral simulation was performed in Xilinx Vivado Design Suite to evaluate the fixed-point quantization error. SoC estimation data are exported into MATLAB and compared with the US06 SoC data, computing an Absolute Error (AE) as

$$AE = abs(SOC_{US06} - SOC_{estimated}) \tag{6}$$

The AE of the HDL-coded SVM was compared with that achieved with a MATLAB SVM simulation performed in double precision. The results are shown in Figure 2, where a maximum error of 3.1% can be inferred. From the figure, MATLAB and Vivado (behavioral simulation) SoC estimations cannot be easily distinguished from each other. For this reason, an enlargement is reported: an error of approximately 10^{-5} between the two approaches was observed.

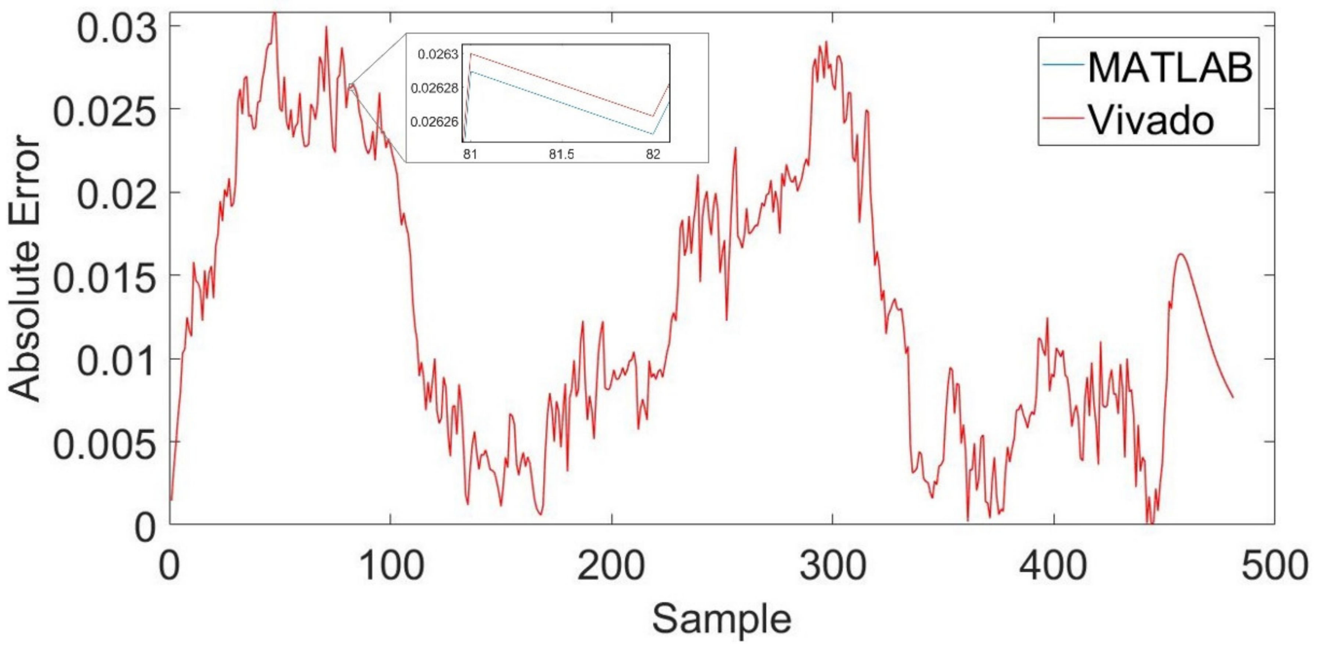

Figure 2. SoC estimation AE in MATLAB floating point-based simulations and Vivado fixed-point-based behavioral simulations.

Next, area occupation and timing performance were evaluated through post-implementation simulations. In Table 4, the occupied area is reported.

Table 4. Area occupation comparison with other studies.

	FPGA	Slice LUTs Utilization (%)	Slice Registers (%)	DSP Slices (%)
Proposed	Artix 7	880/63,400 (1.39%)	300/126,800 (0.24%)	0/240 (0%)
[20]	Artix 7	14,427/63,400 (22.76%)	196/126,800 (0.15%)	8/240 (3.33%)
[41]	Virtex 7	1123/303,600 (1%)	751/607,200 (1%)	1125/2800 (40%)

Compared with other FPGA implementations of NN-based SOC estimations [43], very limited resources are required. Hence, the presented approach allows the placing of multiple instances of the proposed algorithm on the same FPGA. This can be useful when several cells have to be monitored in parallel (e.g., in a battery pack case). For example, the aforementioned case of 60 cells in series could be easily managed by this solution. By comparison, the ECM model presented in [21] occupied 23% of the whole FPGA, preventing more than four instances from being programmed on the same platform. Moreover, unlike in [43], no DSP slices were used, allowing the use of smaller and low-cost devices as well.

Considering a clock frequency of 100 MHz, the post-implementation timing performance resulted in Worst Negative Slack = 0.188 ns, Worst Hold Slack = 0.054 ns, and Worst Pulse Width Slack = 4.5 ns.

Finally, the whole system was tested after programming the FPGA device. Thanks to the VHDL-coded UART module, the FPGA communicates with the MATLAB suite and receives the US06 drive cycle test data. Next, the FPGA processes the input data and estimates the SoC, sending back the results to MATLAB. Finally, the on-board processed output is compared with the expected one, as shown in Figure 3.

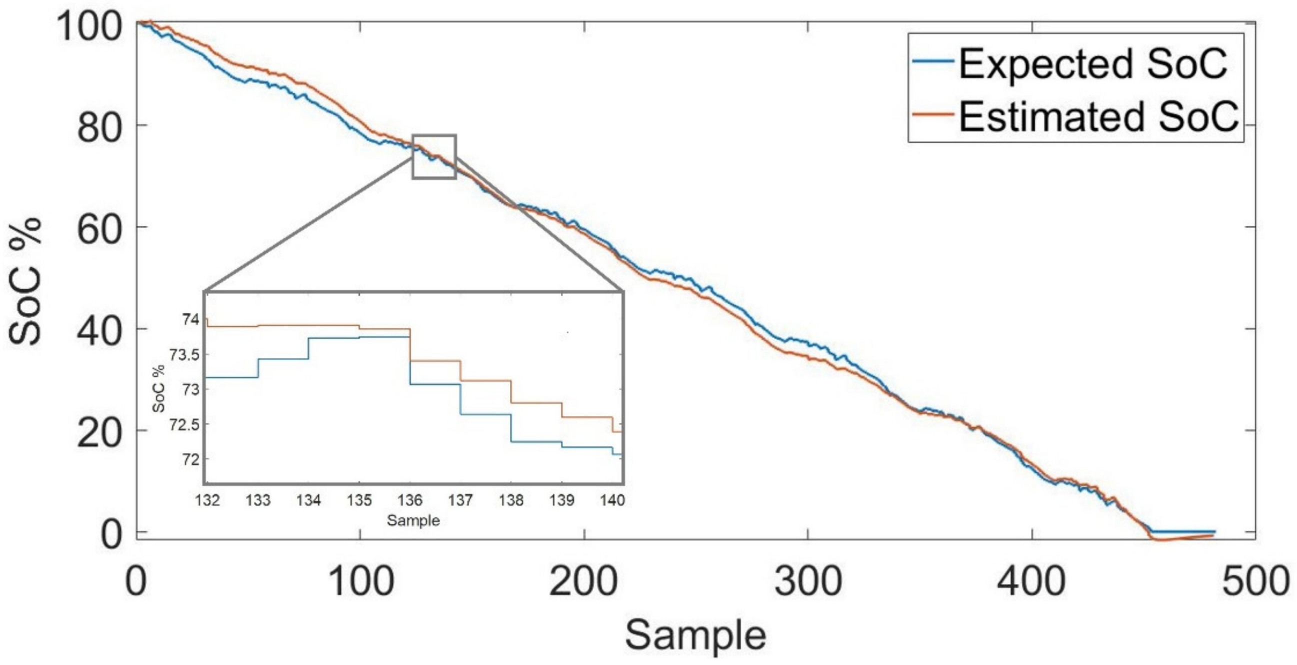

Figure 3. Comparison of expected SoC with SoC processed data on the FPGA board.

It can be seen that FPGA computations (red line) can track changes in the expected battery SoC (blue line), achieving an RMSE of 1.4% and an MAE of 1.2%. Moreover, a determination coefficient [37] R^2 of 99.7% and a maximum error of 3.1% are computed. Compared to the behavioral simulation, the estimation in the final implementation did not worsen; hence, these SoC evaluations completely overlapped.

The accuracy estimation of the proposed ACO-optimized SVR model was compared with the cell model-based approach [21] in Table 5. In this case, two different implementations were considered. In the first implementation, a constant parameters solution was analyzed, i.e., mean values of extracted ECM parameters were assumed. Next, a variable parameters approach was used, i.e., each ECM parameter value was stored in a Look-Up-Table indexed by the current estimated SoC [21].

Table 5. Comparison of performance in SoC estimation through SVR approach, model-based approach [21] with constant parameters, and the same approach [21] with SoC-varying parameters.

Approach	RMSE (%)	Max Error (%)	MAE (%)
ACO-SVR	1.4	3.1	1.2
[21] const. par.	1.5	4.3	1.1
[21] var. par.	2.5	5.4	2.2

The ACO-SVR model outperforms both cell model-based approaches in terms of estimation accuracy. Even if the MAE is slightly better, when using constant parameters [21], the SVR results in better RMSE and a lower maximum percentage error on each estimation. This may be due to the loss of accuracy in parameter estimation when the battery cell is almost fully charged or discharged. Thus, the data-driven approach overcomes the offline parameter estimation problem, even providing a good SoC assessment.

Furthermore, the FPGA's circuit performance was compared with other studies. Notably, using a test set that is different from the training set can produce a general worsening in the error metrics. Nevertheless, the performances were good, as shown in Table 6.

Table 6. Comparison of error between estimated and expected SoC (proposed model implemented on FPGA vs. other studies).

Approach	Kernel	RMSE (%)	Max Error (%)	MAE (%)	R^2 (%)
ACO-SVR	Linear	1.4	3.1	1.2	99
[35]	Quadratic	2.5	13	-	-
[37]	RBF	-	-	-	97
[36]	RBF	-	1.5	1.2	-

In particular, since, to the best of our knowledge no FPGA implementations of SVR applied to SoC are reported in other studies, a microcontroller-based implementation [35] was considered. In our model, the performance greatly improved. In addition, with the proposed implementation, it is possible to exploit the advantages due to the intrinsic parallelism of the FPGA, as discussed previously. In [37], a Labview-based SVM implementation was reported and validated with new data not used for the training. In [36], the same MAE was achieved with a lower maximum error, in a PC-based implementation. However, the ACO-SVR results were obtained on different datasets, based on measured battery data, and by performing an estimation using FPGA hardware. This indicates the feasibility of the approach in real environments, where PCs might not be available. Moreover, the results were achieved with a lower-complexity kernel, which allows an easier hardware implementation and lower resource utilization.

4. Conclusions

In this paper, the use of SVM as a regression method to estimate a battery cell SoC is presented. A lo- complexity linear kernel combined with an ACO algorithm allowed the training of a simple but accurate SVR model. This proposed solution allows the estimation of SoC without the need to model the particular cell, making it a more general approach, suitable for different battery cells and requiring a shorter development time. The design of the SVR model requires a huge amount of measured data, both for the model training and the test phase. In this study, we referred to some large real battery measurement datasets that have been provided in other studies [25–27]. The obtained SVR model was implemented on a Xilinx Artix 7 FPGA, demonstrating low area occupation with respect to other solutions reported in relevant published research. This creates the possibility of implementing multiple instances of the same algorithm working in parallel to manage multiple cells, for example, in a battery pack framework.

To train the ACO-optimized SVR model, a large dataset [25] for Panasonic 18650 Li-Ion battery cells was used. The developed model was coded in Hardware Description Language and then programmed into the selected FPGA device by avoiding the use of DSP slices, to improve portability on different FPGAs. The fixed-point architecture was designed on a signed 32-bit word length data size with 23 fractional bits. The designed SVR model was then 10-fold cross-validated and also tested over a US06 driving cycle dataset [25] to prove its generalization capability when working on real environment data.

The proposed solution was compared with a model-based circuit also implemented on the same FPGA [21]. The error metrics and area occupation of our data-driven approach outperformed the ECM solution. Furthermore, our solution was compared with those of other studies, demonstrating similar performance to PC-based solutions. Moreover, considering embedded approaches, our work was compared with a microcontroller-based implementation, resulting in an improved overall system performance.

It can be concluded that the proposed approach combines acceptable errors with a good performance in terms of area occupation. In the future, even though Li-Ion is one of the most widely used and promising technologies, the same algorithm could be easily applied in battery cells based on different chemistries (e.g., $LiCoO_2$-, $LiFePO_4$- or Nickel-based batteries), since none of the required inputs is technology-dependent. Finally, the management of many battery cells in parallel, exploiting the lower area occupation, flexibility, and intrinsic parallelism of the FPGA device, should be investigated.

Author Contributions: Conceptualization, M.S., V.B. and I.D.M.; methodology, M.S., V.B. and I.D.M.; software M.S.; validation, M.S., V.B. and I.D.M.; formal analysis, M.S., V.B. and I.D.M.; investigation, M.S. and V.B.; data curation, M.S., V.B. and I.D.M.; writing—original draft preparation, M.S., V.B. and I.D.M.; writing—review and editing, M.S., V.B. and I.D.M.; visualization, M.S., V.B. and I.D.M.; supervision, I.D.M.; funding acquisition, I.D.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research has financially been supported by the Project "Advanced Approaches for batteries State of Charge (SoC) evaluation", funded in 2021 by Programme "FIL-Quota Incentivante" of University of Parma and co-sponsored by Fondazione Cariparma.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: Kollmeyer, P. Panasonic 18650PF Li-ion Battery Data. Available online: https://data.mendeley.com/datasets/wykht8y7tg/1.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sathiyanarayanan, J.S.; Kumar, A.S. Maximization battery lifetime and improving efficiency. In Proceedings of the 2012 International Conference on Devices, Circuits and Systems, ICDCS 2012, Tamil Nadu, India, 15–16 March 2012; pp. 603–606.
2. Rong, D.; Yang, B.; Chen, C. Model Predictive Climate Control of Electric Vehicles for Improved Battery Lifetime. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 5457–5462.
3. Yong, J.Y.; Ramachandaramurthy, V.K.; Tan, K.M.; Mithulananthan, N. A review on the state-of-the-art technologies of electric vehicle, its impacts and prospects. *Renew. Sustain. Energy Rev.* **2015**, *49*, 365–385. [CrossRef]
4. Artakusuma, D.D.; Afrisal, H.; Cahyadi, A.I.; Wahyunggoro, O. Battery management system via bus network for multi battery electric vehicle. In Proceedings of the 2014 International Conference on Electrical Engineering and Computer Science, ICEECS 2014, Kuta, Indonesia, 24–25 November 2014; pp. 179–181.
5. Lan, C.W.; Lin, S.S.; Syue, S.Y.; Hsu, H.Y.; Huang, T.C.; Tan, K.H. Development of an intelligent lithium-ion battery-charging management system for electric vehicle. In Proceedings of the 2017 IEEE International Conference on Applied System Innovation: Applied System Innovation for Modern Technology, ICASI 2017, Sapporo, Japan, 13–17 May 2017; pp. 1744–1746.
6. Kim, M.J.; Chae, S.H.; Moon, Y.K. Adaptive Battery State-of-Charge Estimation Method for Electric Vehicle Battery Management System. In Proceedings of the International SoC Design Conference, ISOCC 2020, Yeosu, Korea, 21–24 October 2020; pp. 288–289.
7. Alvarez, J.M.; Sachenbacher, M.; Ostermeier, D.; Stadlbauer, H.J.; Hummitzsch, U.; Alexeev, A. D6.1—Analysis of the State of the Art on BMS. Technical Report. 2017. Available online: http://everlasting-project.eu/wp-content/uploads/2016/11/EVERLASTING_D6.1_final_20170228.pdf (accessed on 23 September 2021).
8. How, D.N.T.; Hannan, M.A.; Lipu, M.S.H.; Ker, P.J. State of Charge Estimation for Lithium-Ion Batteries Using Model-Based and Data-Driven Methods: A Review. *IEEE Access* **2019**, *7*, 136116–136136. [CrossRef]
9. Plett, G.L. Efficient Battery Pack State Estimation using Bar-Delta Filtering. In Proceedings of the 24th International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium and Exhibition 2009, EVS 24, Stavanger, Norway, 13–16 May 2009; Volume 1, pp. 163–170.
10. Zhang, R.; Xia, B.; Li, B.; Cao, L.; Lai, Y.; Zheng, W.; Wang, H.; Wang, W. State of the art of lithium-ion battery SOC estimation for electrical vehicles. *Energies* **2018**, *11*, 1820. [CrossRef]
11. Espedal, I.B.; Jinasena, A.; Burheim, O.S.; Lamb, J.J. Current trends for state-of-charge (SoC) estimation in lithium-ion battery electric vehicles. *Energies* **2021**, *14*, 3284. [CrossRef]
12. He, H.; Xiong, R.; Fan, J. Evaluation of lithium-ion battery equivalent circuit models for state of charge estimation by an experimental approach. *Energies* **2011**, *4*, 582–598. [CrossRef]
13. Deng, D.; Wang, S.; Chen, L. An improved second-order electrical equivalent modeling method for the online high power Li-ion battery state of charge estimation. In Proceedings of the 12th IEEE Energy Conversion Congress & Exposition-Asia (ECCE-Asia), Singapore, 24–27 May 2021; pp. 1725–1729.
14. Saboo, K.; Mangsule, R.; Deshpande, A.S. State of Charge (SoC) Estimation of Li-Ion Battery. In Proceedings of the International Conference on Emerging Smart Computing and Informatics, ESCI 2021, Pune, India, 5–7 March 2021; pp. 340–345.
15. Wang, L.; Wang, L.; Liao, C. Research on improved EKF algorithm applied on estimate EV battery SOC. In Proceedings of the Asia-Pacific Power and Energy Engineering Conference, APPEEC, Chengdu, China, 28–31 March 2010; pp. 1–4.

16. Zhang, F.; Ur Rehman, M.M.; Wang, H.; Levron, Y.; Plett, G.; Zane, R.; Maksimović, D. State-of-charge estimation based on microcontroller-implemented sigma-point Kalman filter in a modular cell balancing system for Lithium-Ion battery packs. In Proceedings of the 2015 IEEE 16th Workshop on Control and Modeling for Power Electronics (COMPEL), Vancouver, BC, Canada, 12–15 July 2015; pp. 1–7.
17. Johnsema, B.; Janakiraman, K. Reliable SOC estimation for battery powered embedded system. In Proceedings of the International Conference on Information Communication and Embedded Systems (ICICES2014), Chennai, India, 27–28 February 2014; pp. 1–5.
18. Monmasson, E.; Idkhajine, L.; Naouar, M.W. FPGA-based controllers. *IEEE Ind. Electron. Mag.* **2011**, *5*, 14–26. [CrossRef]
19. Otero, N.; Rahimi-Eichi, H.; Rodriguez-Andina, J.J.; Chow, M.Y. FPGA implementation of an observer for state of charge estimation in lithium-polymer batteries. In Proceedings of the International Conference on Mechatronics and Control (ICMC), Jinzhou, China, 3–5 July; 2014; pp. 1646–1651.
20. Morello, R.; Di Rienzo, R.; Roncella, R.; Saletti, R.; Baronti, F. Hardware-in-the-loop platform for assessing battery state estimators in electric vehicles. *IEEE Access* **2018**, *6*, 68210–68220. [CrossRef]
21. Stighezza, M.; Bianchi, V.; De Munari, I. HDL Code Generation from SIMULINK Environment for Li-Ion Cells State of Charge and Parameter Estimation. *Lect. Notes Electr. Eng.* **2021**, *738*, 136–143.
22. Zhang, R.; Xia, B.; Li, B.; Cao, L.; Lai, Y.; Zheng, W.; Wang, H.; Wang, W.; Wang, M. A study on the open circuit voltage and state of charge characterization of high capacity lithium-ion battery under different temperature. *Energies* **2018**, *11*, 2408. [CrossRef]
23. Zhang, D.Y.; Ma, J.; Zhang, K. State of Charge Estimation for Battery Based on Improved Cubature Kalman Filter. In Proceedings of the 20th COTA International Conference of Transportation Professionals: Advanced Transportation Technologies and Development-Enhancing Connections, CICTP 2020, Xi'an, China, 14–16 August 2021; pp. 2303–2316.
24. Danko, M.; Adamec, J.; Taraba, M.; Drgona, P. Overview of batteries State of Charge estimation methods. In Proceedings of the13th International Scientific Conference On Suistanable, modern and safe transport (TRANSCOM 2019), Novy Smokovec, Slovak Republic, 29–31 May 2019; pp. 186–192.
25. Kollmeyer, P. Panasonic 18650PF Li-ion Battery Data. Available online: https://data.mendeley.com/datasets/wykht8y7tg/1 (accessed on 23 September 2021).
26. Bole, B.; Kulkarni, C.; Daigle, M. Randomized Battery Usage Data Set. Available online: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/ (accessed on 1 September 2021).
27. Kollmeyer, P.; Vidal, C.; Naguib, M.; Skells, M. LG 18650HG2 Li-ion Battery Data and Example Deep Neural Network xEV SOC Estimator Script. Available online: https://data.mendeley.com/datasets/cp3473x7xv/3 (accessed on 1 October 2021).
28. Vidal, C.; Malysz, P.; Kollmeyer, P.; Emadi, A. Machine Learning Applied to Electrified Vehicle Battery State of Charge and State of Health Estimation: State-of-the-Art. *IEEE Access* **2020**, *8*, 52796–52814. [CrossRef]
29. Song, X.; Yang, F.; Wang, D.; Tsui, K.L. Combined CNN-LSTM Network for State-of-Charge Estimation of Lithium-Ion Batteries. *IEEE Access* **2019**, *7*, 88894–88902. [CrossRef]
30. Kang, L.W.; Zhao, X.; Ma, J. A new neural network model for the state-of-charge estimation in the battery degradation process. *Appl. Energy* **2014**, *121*, 20–27. [CrossRef]
31. Tong, S.; Lacap, J.H.; Park, J.W. Battery state of charge estimation using a load-classifying neural network. *J. Energy Storage* **2016**, *7*, 236–243. [CrossRef]
32. Trinandana, G.A.; Pratama, A.W.; Prasetyono, E.; Anggriawan, D.O. Real Time State of Charge Estimation for Lead Acid Battery Using Artificial Neural Network. In Proceedings of the 2020 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 22–23 July 2020; pp. 363–368.
33. Chemali, E.; Kollmeyer, P.J.; Preindl, M.; Emadi, A. State-of-charge estimation of Li-ion batteries using deep neural networks: A machine learning approach. *J. Power Sources* **2018**, *400*, 242–255. [CrossRef]
34. Yang, F.; Song, X.; Xu, F.; Tsui, K.L. State-of-Charge Estimation of Lithium-Ion Batteries via Long Short-Term Memory Network. *IEEE Access* **2019**, *7*, 53792–53799. [CrossRef]
35. Hansen, T.; Wang, C.J. Support vector based battery state of charge estimator. *J. Power Sources* **2005**, *141*, 351–358. [CrossRef]
36. Li, R.; Xu, S.; Li, S.; Zhou, Y.; Zhou, K.; Liu, X.; Yao, J. State of charge prediction algorithm of lithium-ion battery based on PSO-SVR cross validation. *IEEE Access* **2020**, *8*, 10234–10242. [CrossRef]
37. Antón, J.C.Á.; Nieto, P.J.G.; de Cos Juez, F.J.; Lasheras, F.S.; Vega, M.G.; Gutiérrez, M.N.R. Battery state-of-charge estimator using the SVM technique. *Appl. Math. Model.* **2013**, *37*, 6244–6253. [CrossRef]
38. Osornio-Rios, R.A.; Romero-Troncoso, R.D.J.; Morales-Velazquez, L.; De Santiago-Perez, J.J.; Rivera-Guillen, R.D.J.; Rangel-Magdaleno, J.D.J. A real-time FPGA based platform for applications in mechatronics. In Proceedings of the 2008 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 3–5 December 2008; pp. 289–294.
39. Al-Mahmood, A.; Opoku, M. A Study of FPGA-based System-on-Chip Designs for Real-Time Industrial Application. *Int. J. Comput. Appl.* **2017**, *163*, 9–19. [CrossRef]
40. Kilic, A.; Koroglu, S.; Demircali, A.; Kesler, S.; Oner, Y.; Karakas, E.; Sergeant, P. Design of Master and Slave Modules on Battery Management System for Electric Vehicles. In Proceedings of the 6th International Conference on Advanced Technology & Sciences (ICAT'Riga), Riga, Latvia, 12–15 September 2017; pp. 161–166.
41. Ruiz-Llata, M.; Yébenes-Calvino, M. FPGA implementation of support vector machines for 3D object identification. *Lect. Notes Comput. Sci.* **2009**, *5768*, 467–474.

42. Ruiz-Llata, M.; Guarnizo, G.; Yébenes-Calvino, M. FPGA implementation of a support vector machine for classification and regression. In Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 18–23 July 2010; pp. 1–5.
43. Kumar, B.; Khare, N.; Chaturvedi, P.K. FPGA-based design of advanced BMS implementing SoC/SoH estimators. *Microelectron. Reliab.* **2018**, *84*, 66–74. [CrossRef]
44. Dorigo, M.; Blum, C. Ant colony optimization theory: A survey. *Theor. Comput. Sci.* **2005**, *344*, 243–278. [CrossRef]
45. Zheng, L.; Yu, M.; Yu, S. Support vector regression and ant colony optimization for combustion performance of boilers. In Proceedings of the 2008 Fourth International Conference on Natural Computation, Jinan, China, 18–20 October 2008; pp. 178–182.
46. Hong, W.C.; Dong, Y.; Zheng, F.; Lai, C.Y. Forecasting urban traffic flow by SVR with continuous ACO. *Appl. Math. Model.* **2011**, *35*, 1282–1291. [CrossRef]
47. Wang, H.; Yang, S. Electricity Consumption Prediction Based on SVR with Ant Colony Optimization. *Telkomnika Indones. J. Electr. Eng.* **2013**, *11*, 6928–6934. [CrossRef]
48. Xilinx Inc. 7 Series FPGAs Data Sheet: Overview (DS180). Available online: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf (accessed on 23 September 2021).
49. Cortes, C.; Vapnik, V. Support-Vector Networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
50. Flake, G.W.; Lawrence, S. Efficient SVM regression training with SMO. *Mach. Learn.* **2002**, *46*, 271–290. [CrossRef]
51. Fletcher, T. Support Vector Machines Explained. Available online: https://www.csd.uwo.ca/~{}xling/cs860/papers/SVM_Explained.pdf (accessed on 23 September 2021).
52. Wang, H.; Hu, D. Comparison of SVM and LS-SVM for Regression. In Proceedings of the 2005 International Conference on Neural Networks and Brain, Beijing, China, 13–15 October 2005; pp. 279–283.
53. Mathworks Fitrsvm MATLAB Function. Available online: https://it.mathworks.com/help/stats/fitrsvm.html (accessed on 23 September 2021).
54. US06 Drive Cycle. Available online: https://dieselnet.com/standards/cycles/ftp_us06.php (accessed on 23 September 2021).
55. Bhatti, A.H.U.; Kazmi, S.A.A.; Tariq, A.; Ali, G. Development and analysis of electric vehicle driving cycle for hilly urban areas. *Transp. Res. Part D Transport. Environ.* **2021**, *99*, 103025. [CrossRef]
56. Li, J.; Ye, M.; Meng, W.; Xu, X.; Jiao, S. A Novel State of Charge Approach of Lithium Ion Battery Using Least Squares Support Vector Machine. *IEEE Access* **2020**, *8*, 195398–195410. [CrossRef]
57. Mahmoodi, D.; Soleimani, A.; Khosravi, H.; Taghizadeh, M. FPGA Simulation of Linear and Nonlinear Support Vector Machine. *J. Softw. Eng. Appl.* **2011**, *4*, 320–328. [CrossRef]
58. Batista, G.C.; Oliveira, D.L.; Saotome, O.; Silva, W.L.S. A low-power asynchronous hardware implementation of a novel SVM classifier, with an application in a speech recognition system. *Microelectron. J.* **2020**, *105*, 1–17. [CrossRef]
59. Alobaedy, M.M.; Khalaf, A.A.; Muraina, I.D. Analysis of the number of ants in ant colony system algorithm. In Proceedings of the 2017 5th International Conference on Information and Communication Technology (ICoIC7), Melaka, Malaysia, 17–19 May 2017; pp. 1–5.
60. Panasonic Lithium Ion NCR18650PF Datasheet. Available online: https://na.industrial.panasonic.com/products/batteries/rechargeable-batteries/lineup/lithium-ion/series/90729/model/90730 (accessed on 23 September 2021).
61. EEMB Lithium-Ion Battery LIR18650 2600mAh Datasheet. Available online: https://www.eemb.com/model/lir18650(2600).html (accessed on 23 September 2021).