Briliant: Jurnal Riset dan Konseptual
Vol. 9, No. 3, Agustus 2024 pp. 729-739
E-ISSN: 2541-4224, P-ISSN: 2541-4216
DOI: http://dx.doi.org/10.28926/briliant.v9i3.1874

Estimasi *State-of-Charge* Pada Baterai *Lithium-Ion* Menggunakan *Deep Neural Network*

Haniifan Patra Amrullah[(1)], Novie Ayub Windarko[(2)], Bambang Sumantri[(3)]

Politeknik Elektronika Negeri Surabaya,
JL. Raya ITS, Keputih, Kec. Sukolilo, Kota SBY, Jawa Timur, Indonesia

Email: [1]haniifan.patra27@gmail.com, [2]ayub.sch@gmail.com, [3]bambang@pens.ac.id

Tersedia Online di
http://www.jurnal.unublitar.ac.id/index.php/briliant

Sejarah Artikel
Diterima 4 April 2024
Direvisi 9 Juni 2024
Disetujui 10 Juni 2024
Dipublikasikan 31 Agutus 2024

Keywords:
State-of-Charge, Deep Neural Network, Battery

Kata Kunci:
State-of-Charge, Deep Neural Network, Baterai

***Corresponding Author*:**
Name:
Haniifan Patra Amrullah
Email:
haniifan.patra27@gmail.com

Abstract*: As electric vehicles (EV) become increasingly popular in the automotive world, an accurate State-of-Charge (SoC) estimation is critical to optimizing energy utilization, increasing driving range and ensuring long-lasting battery system. This research focuses on the application of Deep Neural Networks (DNN) as an SoC estimation method in EV, exploiting the inherent capacity of DNN to learn complex relationships in vast data sets. The results of the performed simulations show that the proposed DNN-based SoC estimation method achieves a high level of accuracy, outperforming traditional estimation techniques, especially in scenarios involving rapid changes in driving conditions. This research also explores the impact of Neural Networks architecture and hyperparameter tuning on overall performance and provides insights for optimizing DNN-based SoC estimation systems. From the tests that have been carried out, an error value of 1.3% is obtained from the results of the training carried out on the DNN structure that has been prepared.*

Abstrak: Seiring dengan semakin populernya kendaraan listrik (EV) di dunia otomotif, estimasi *State-of-Charge* (SoC) yang akurat sangat penting untuk mengoptimalkan pemanfaatan energi, meningkatkan jangkauan berkendara dan memastikan sistem baterai tahan lama. Penelitian ini berfokus pada penerapan *Deep Neural Networks* (DNN) sebagai metode estimasi SoC pada EV, memanfaatkan kapasitas bawaan DNN untuk mempelajari hubungan kompleks dalam kumpulan data yang luas. Hasil dari simulasi yang dilakukan menunjukkan bahwa metode estimasi SoC berbasis DNN yang diusulkan mencapai tingkat akurasi yang tinggi, mengungguli teknik estimasi tradisional, terutama dalam skenario yang melibatkan perubahan cepat dalam kondisi berkendara. Penelitian ini juga mengeskplorasi dampak arsitektur *Neural Networks* dan penyetelan *hyperparameter* terhadap kinerja secara keseluruhan, serta memberikan wawasan untuk mengoptimalkan sistem estimasi SoC berbasis DNN. Dari pengujian yang telah dilakukan didapat nilai *error* sebesar 1.3% dari hasil *training* yang dilakukan terhadap struktur DNN yang telah disusun.

PENDAHULUAN

Pada beberapa tahun terakhir, perubahan iklim yang terjadi, yang dikarenakan penggunaan energi fosil yang berkepanjangan sebagai sumber energi menyebabkan kekhawatiran tersendiri dalam masyarakat. Semakin berkurangnya persediaan sumber energi fosil tersebut juga menjadi salah satu faktor yang menggeser perhatian beberapa pemerintahan dunia untuk beralih ke sumber energi alternatif yaitu sumber energi terbarukan (Adebayo dkk., 2023). Selama periode 2018 dan 2023 sumber energi terbarukan berkontribusi sekitar 70% dari peningkatan produksi energi listrik secara global. Sesuai dengan *International Energy Agency* (IEA), penggunaan sumber daya energi terbarukan akan meningkat sebesar 11% hingga 15% antara tahun 2008 dan 2035 (- International Energy Agency, t.t.).

Maraknya penggunaan sumber energi terbarukan sebagai upaya untuk mengurangi jejak karbon di atmosfir, pergeseran untuk beralih dari kendaraan dengan *Intercal Combustion Engine* (ICE) pada *Electric Vehicle* (EV) juga semakin didorong oleh beberapa negara (Hoeft, 2021). Inggris merupakan salah satu negara yang menetapkan larangan penjualan ICE baru mulai tahun 2035 (Xia dkk., 2022). Dengan semakin bertumbuhnya kebutuhan produksi untuk EV, juga mendorong penelitian dan pengembangan pada EV degan pesat (Barman dkk., 2023).

Untuk EV sendiri, baterai merupakan komponen utama yang menjadi sumber energi EV (Shu dkk., 2020). Dalam upaya untuk menjaga agar baterai dapat bertahan lama, penelitian pada Battery Management System (BMS) terus dilakukan untuk mendapat hasil yang baik (Lebrouhi dkk., 2021). Diantara salah satu komponen BMS, salah satu komponen yang penting adalah estimasi State of Charge (SoC). SoC adalah jumlah sisa daya yang tersisa pada baterai (Zhang & Fan, 2020); untuk mendapat nilai tersebut etimasi dilakukan. Keakuratan dalam melakukan estimasi SoC menjadi hal yang sangat penting untuk bisa menjaga umur baterai agar bisa bertahan lama (Hossain Lipu dkk., 2020). Hasil estimasi yang akurat, dapat mencegah baterai untuk berada pada kondisi overcharge dan atau overdischarge, dengan demikian dapat memperpanjang umur baterai.

Salah satu cara untuk melakukan estimasi SoC tersebut adalah dengan menggunakan Machine Learning (ML) sebagai metodenya. Menggunakan ML sebagai metode untuk mendapat nilai estimasi SoC, nilai akurasi yang tinggi bisa didapatkan tanpa perlu memodelkan baterai dengan rangkaian ekivalen yang dimana untuk mendapatkan rangkaian ekivalen tersebut diperlukan pemahaman mendalam terhadap komposisi baterai. Hal ini mempermudah pekerjaan dan mempersingkat waktu. Sebagai pengganti rangkaian ekivalen, data dari percobaan yang dilakukan pada baterai dibutuhkan untuk nantinya digunakan dalam melakukan training model ML.

Berbagai metode untuk melakukan estimasi SoC pada baterai ini telah banyak dilakukan, salah satunya yang dilakukan oleh Yang, Li, Mao pada 2019 dijelaskan penggunaan Gared Recurrent Unit (GRU) pada Recurrent Neural Network (RNN) untuk melakukan estimasi SoC. GRU disini berfungsi dengan cara menerapkan reset gate dan update gate untuk menentukan apa yang harus dilupakan dan apa yang harus diingat. Menggunakan gating mechanism, reset gate akan menghilangkan informasi yang tidak diinginkan dan melakukan update gate dengan informasi penting yang didapat dari hidden states sebelumnya; dengan kata lain GRU disini mengidentifikasikan informasi penting untuk memperkirakan status saat ini dan karena itu mampu menangani ketergantungan jangka panjang (Yang dkk., 2019).

Lalu pada penelitian yang ditulis oleh Lipu, Hannan, Hussain, Saad, Ayob, Uddin pada tahun 2019 dijelaskan Extrem Learning Machine (ELM) model untuk melakukan estimasi SoC. ELM diusulkan sebagai metode untuk menentukan nilai estimasi, karena metode ini memiliki kecepatan estimasi yang cepat, kinerja generalisasi yang baik dan akurasi tinggi. Untuk menutupi kekurangan ELM yang dimana sangat bergantung pada akurasi training dan jumlah neuron pada hidden layer, Gravitational Search Algorithm (GSA) diterapkan untuk meningkatkan kecerdasan komputasi ELM dengan mencari nilai neuron dan hidden layer yang optimal (Hossain Lipu dkk., 2019).

Sedangkan pada penelitian yang disusun oleh Ningrum, Windarko, Suharningsih pada tahun 2021 dijelaskan penggunaan Modified Coulomb Counting Method dengan Open Circuit Compensation untuk melakukan estimasi SoC. Untuk mendapa tnilai tegangan baterai sebenarnya, uji coba pada baterai dilakukan dengan menggunakan berbagai arus konstan dengan kondisi *Open Circuit Voltage* (OCV) pada baterai selama 1 jam dimana OCV pada baterai berbeda-beda. Sehingga level tegangan baterai tidak bisa disamakan antara baterai yang satu dan yang digunakan. Rerata error yang didapat dengan metode ini berada pada nilai 1% (Ningrum & Windarko, 2021).

METODE

Deep Neural Network

Deep Neural Network (DNN) merupakan salah satu jenis dari *Artificial Neural Network* (ANN). DNN dibangun dengan berdasarkan cara kerja dari *Feed-Forward Neural Network* (FNN), dimana untuk kedalaman layernya desesuaikan agar bisa diklasifikasikan sebagai DNN (Jiang dkk., 2021). Struktur DNN sendiri dalam prinsipnya adalah memodelkan sistem linear dengan cara memetakan keluaran yang diinginkan yang nantinya dapat diamati. Dataset yang digunakan dalam proses *training* didefinisikan sebagai berikut (Chemali dkk., 2018):

$$D = \{(\psi(1), SOC(1)*), (\psi(2), SOC(2)*), \dots, (\psi(\tau), SOC(\tau)*)\} \quad (1)$$

Dimana SOC(t) dan $\psi(t)$ adalah nilai ideal State-of-Charge dan vektor input saat waktu t. untuk vektor masukan itu sendiri itu sendiri didefinisikan sebagai berikut (Chemali dkk., 2018):

$$\psi(t) = [V(t), I(t), T(t), I_{avg}(t), V_{avg}(t)] \quad (2)$$

Dimana V(t), I(t), T(t), Iavg(t), Vavg(t) merepresentasikan tegangan, arus temperatur, arus rata-rata, dan tegangan rata-rata baterai saat waktu t. Arus rata-rata dan tegangan rata-rata keduanya dikalkulasikan selama ξ waktu preseden, dimana berada sekitar 40 sampai 400 time steps. Data ini tidak dikacaukan dengan data total time steps yang didefinisikan dengan τ, dimana $\xi < \tau$.

DNN mempelajari untuk dapat memodelkan perilaku baterai dengan cara melakukan *mapping* masukan untuk menjadi keluaran, dimana keluaran yang diinginkan disini adalah nilai SoC. Untuk dapat menghasilkan keluaran, masukan harus melalui beberapa seri dari matriks perkalian. Misalkan w^l menyatakan *weights* pada *layer* 1, b^l menyatakan bias pada *layer* 1, a^l menyatakan *activation function* pada *layer* 1. Setiap *activation hidden layer, h* dikompputasikan sebagai berikut

$$h_g^l = \sigma\left(\sum_g \left(w_f^l, {}_g h_g^{l-1} + b_g^l\right)\right) \quad (3)$$

Dimana σ adalah *Rectified Linear Unit* (ReLU) *activation function* sebagai berikut:

$$\sigma(x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases} \quad (4)$$

Pada *layer* keluaran, nilai estimasi SoC dihitung secara demikian

$$SOC_k = \eta\left(\sum_g \left(w_f^L, {}_g h_g^{L-1} + b_g^L\right)\right) \quad (5)$$

Dimana L adalah *hidden layer* terakhir dan η sigmoid *activation function* seperti demikian:

$$\eta(x) = \frac{1}{1 + e^{-x}} \quad (6)$$

Pengambilan Data

Data yang digunakan untuk *training* pada penelitian ini menggunakan dataset baterai *lithium-ion* LG 18650 HG2. Untuk mendapatkan data baterai tersebut, peneliti terdahulu melakukan percobaan dengan menggunakan *thermal chamber* dengan 75amp, 5 volt *Digatron Firing Circuit Universal Battery Tester*, dengan akurasi skala penuhnya mencapai 0.1% pada *channel* tegangan dan arus. Percobaan yang dilakukan menggunakan empat *drive cycle* yaitu US06, LA92, UDDS , HWFET. Data baterai dalam dataset tersebut kemudian dinormalisasi agar dapat menyerdehanakan komputasi dan meningkatkan performa *neural network* selama proses *training* dengan mengurangi gangguan skala pada nilai *training* dataset. Normalisasi, pada

prakteknya mampu secara signifikan mengurangi waktu *training* karena nilai *input* akan lebih mudah dikomparasi dan parameter "*learable*" pada *neural network* akan beradaptasi dengan cepat selama proses *training* (Vidal dkk., 2020).

Struktur DNN

Penyusunan algoritma *neural network* mengacu pada algoritma *feed-forward neural network* (FNN) yang dijadikan dasar. Untuk FNN penggunaan algoritma *learning/training* yang paling umum adalah algoritma *back-propagation*(Hemeida dkk., 2020). Pada dasarnya, ini adalah cara untuk memperbarui *weight* jaringan sinaptik dengan menyebarkan kembali vektor gradien dimana setiap elemen didefinisikan sebagai turunan dari ukuran *error* terhadap suatu parameter. Untuk penelitian ini akan menggunakan *Deep Neural Network* dengan lima *input*, dimana *input* tersebut adalah tegangan, arus, temperatur, rata-rata tegangan, rata-rata arus.

$$a_i^{(0)} = x_1 \tag{7}$$

Dimana *i* = 1, 2, 3, 4, 5 menunjukkan fitur *input*.

Pada bagian *hidden layer* menggunakan 5 *layer* dengan *node* berjumlah 81 disetiap *layer*-nya. Mempertimbangkan bahwa jumlah *node* tersebut menghasilkan performa paling baik saat percobaan dengan menggunakan *node* dari 10-100. Untuk *input layer* menuju *hidden layer* pertama, *tanh function* digunakan sebagai *activation function* (Yuen dkk., 2021).

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{8}$$

Dimana *e* adalah basis dari *natural algoritm* dan *x* adalah *input* yang diberikan kepada *tanh function*. *Activation function* yang digunakan pada bagian *hidden layer* adalah *leaky ReLU* (Yuen dkk., 2021).

$$\text{leakyReLU}(x) = \max(alpha \times x, x) \tag{9}$$

Pada persamaan bisa dilihat untuk setiap parameter *x*, *leaky ReLU activation function* akan kembali pada nilai maksimal antara *x* atau *alpha* x *x*, dimana *alpha* adalah nilai konstan positif yang bernilai kecil. Pada peneilitan ini, nilai *Leaky ReLU* yang digunakan bernilai 0.3.
Pada *hidden layer*, untuk setiap *hidden layer l* (dimana *l* = 1, 2, …, *L*) dengan N_l neuron:

$$z_j^{(l)} = \sum_{i=1}^{N_{l-1}} w_{ji}^{(l)} . a_i^{(l-1)} + b_j^{(l)} \tag{10}$$

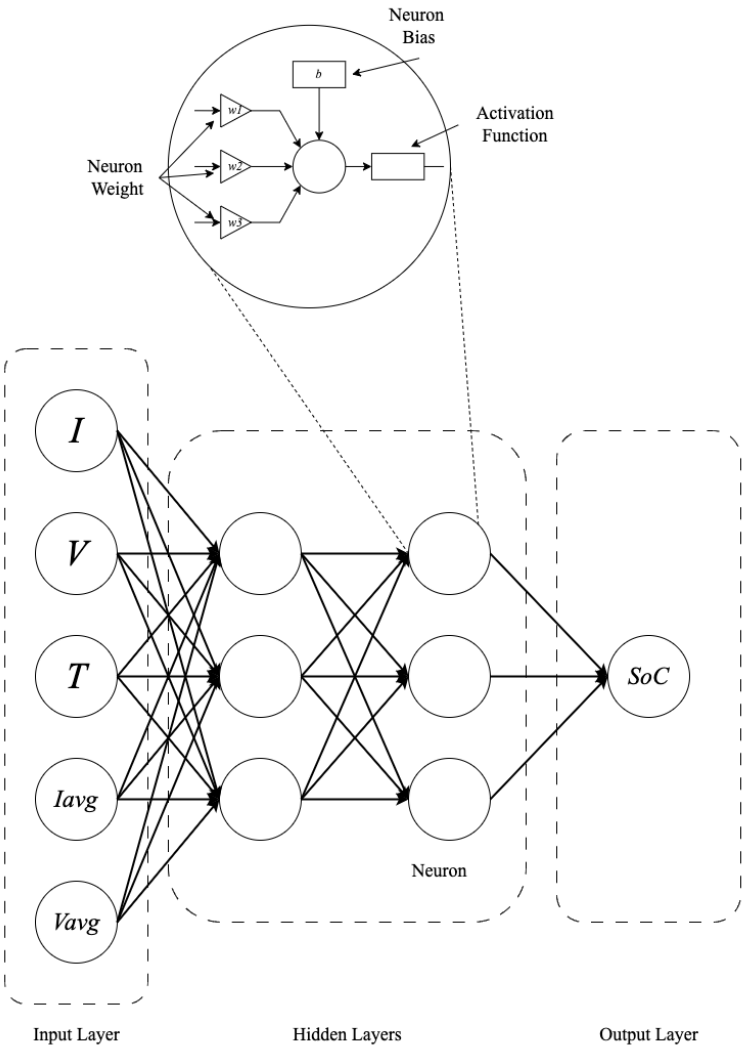
$$a_j^{(l)} = f(z_j^{(l)}) \tag{11}$$

Dimana *j* mengindekskan *neuron* pada *hiden layer*, $w_{ji}^{(l)}$ adalah *weight* yang menyambungkan *neuron i* pada *layer l* – 1 untuk *neuron j* pada *layer l*, $b_j^{(l)}$ adalah bias untuk *neuron j* pada *layer l* dan *f()* adalah *activation function*. Sedangkan untuk *output layer* dijabarkan sebagai berikut:

$$\mathrm{z}_k^{(L+1)} = \sum_{j=1}^{N_L} w_{kj}^{(L+1)} \cdot a_j^{(J)} + b_k^{(L+1)} \tag{12}$$

$$y = a_k^{(L+1)} = f(z_k^{(L+1)} \tag{13}$$

Dimana *k* mengindekskan *neuron* pada *output layer*, $w_{kj}^{(L+1)}$ adalah *weight* yang menghubungkan *neuron j* pada *hidden layer* terakhir pada *output neuron*, $b_k^{(L+1)}$ adalah bias untuk *output neuron* dan *f()* adalah *activation function*.

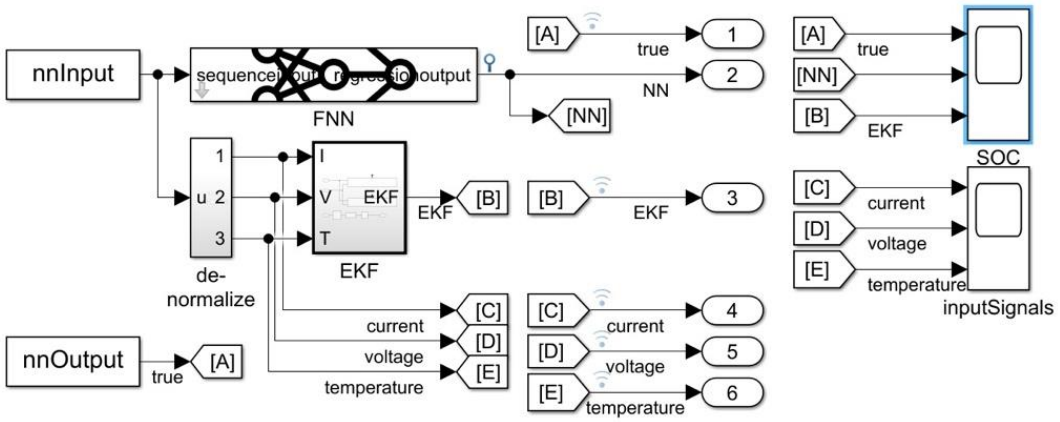


Gambar 1. Diagram DNN

Pada gambar 1 ini ditampilkan diagram susunan struktur DNN. Pada *input layer* terdapat 5 *input* yang digunakan yaitu; *I, V, T,* I_{avg}, dan V_{avg}. Pada *hidden layers* akan dilakukan modifikasi dimana akan ditentukan jumlah *layer* yang digunakan dan jumlah *neuron* pada setiap *hidden layer*. Pada *output layer* adalah keluaran yang diinginkan, yaitu nilai hasil estimasi SoC. Nilai pada tiap *neuron* akan dipengaruhi *neuron weight, neuron bias* dan *activation function*.

Simulasi

Pelaksanaan simulasi dilakukan setelah algorima untuk struktur DNN telah diselesaikan dan bisa dijalankan serta telah mendapatkan hasil yang diinginkan. Percobaan dilakukan melalui *Simulink* yang ada pada MATLAB. Pada simulasi yang dilakukan ditambahkan metode pembanding sebagai referensi apakah hasil yang didapat dari DNN sudah bisa dikatakan baik. Pada simulasi ini metode pembanding yang ditambahkan adalah *Extended Kalman Filter* (EKF). Pada simulasi juga terdapat data hasil observasi yang merupakan data *real* dari SoC.

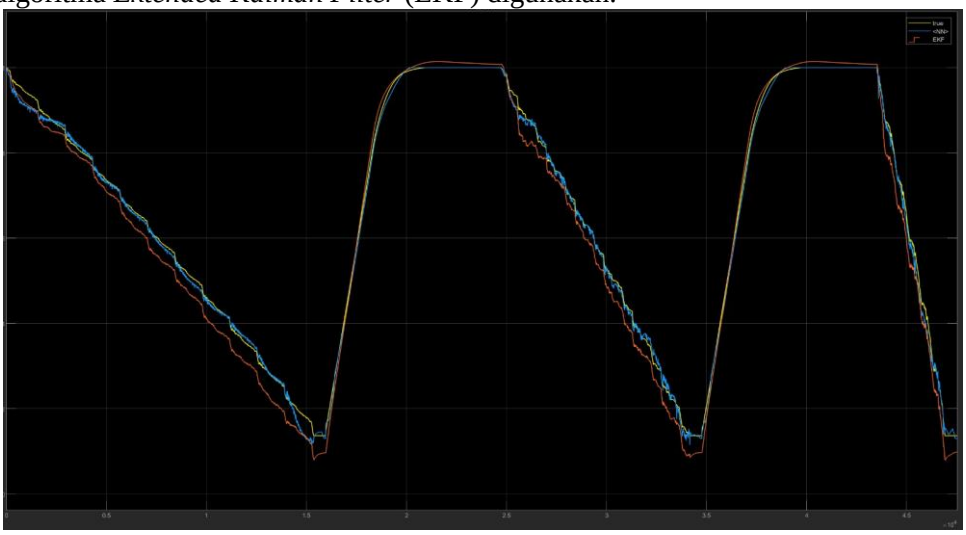


Gambar 2. *Simulink* simulasi sistem estimasi SoC menggunkan DNN

Gambar 2. Merupakan skema *Simulink* yang dibuat untuk melakukan simulasi estimasi SoC menggunakan DNN yang telah dibentuk. Simulasi ini dibuat dengan tujuan untuk melihat hasil prediksi estimasi SoC dari DNN yang telah dibentuk dan juga membandingkan hasill tersebut denagan metode lain, yang dimana metode lain tersebut adalah *Extended Kalman Filter*.

HASIL DAN PEMBAHASAN

Pada bagian ini akan menampilkan pengujian yang dilakukan pada MATLAB Simulink. Pengujian nantinya diharapkan mendapatkan hasil estimasi *State-of-Charge* (SoC). Algoritma *Deep Neural Network* (DNN) diharapkan dapat menghasilkan nilai estimasi dengan *error* yang minimum dan mendekati nilai asli dari data observasi yang didatpakan. *Input* yang digunakan pada DNN ini menggunakan data tegangan, arus, temeratur, rata-rata tegangan, dan rata-rata arus yang didapat dari dataset. Sebagai algoritma pembanding untuk hasil dari estimasi yang dilakukan DNN, algoritma *Extended Kalman Filter* (EKF) digunakan.

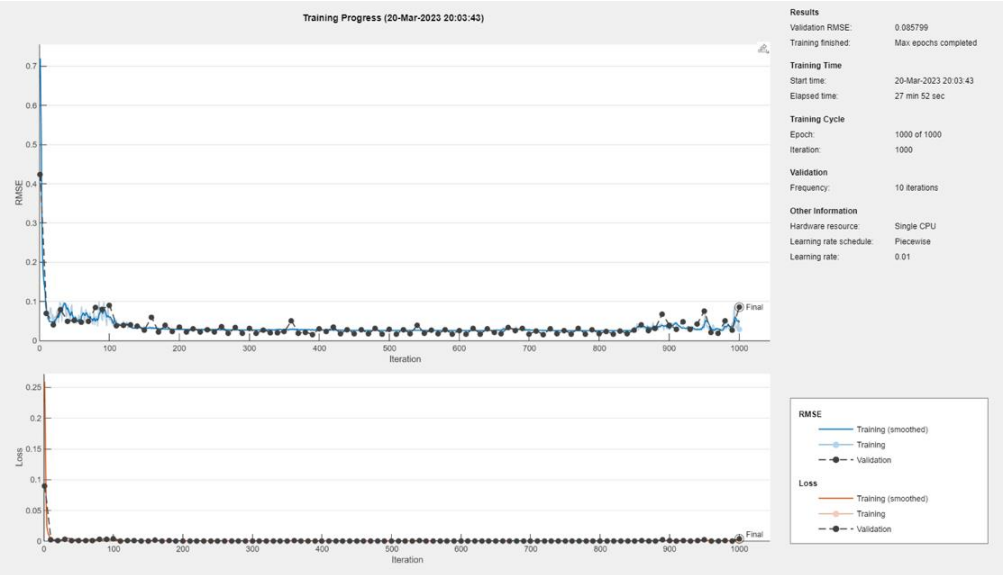


Gambar 3. Hasil simulasi dari *Simulink*

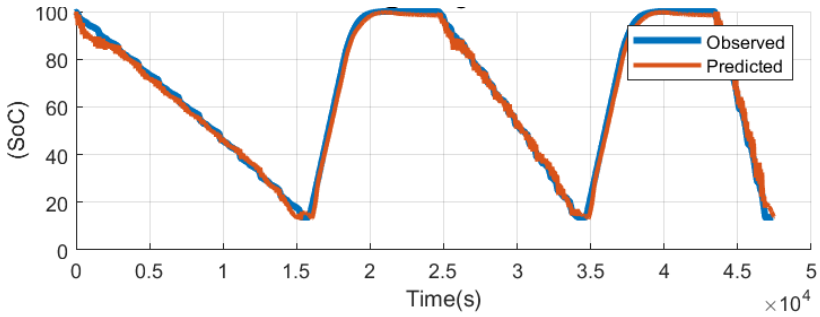
Pada Gambar 3. Adalah hasil dari estimasi yang telah dilakukan DNN yang telah dibentuk. Pada grafik tersebut terdapat data observasi yang dimana data tersebut adalah data *real* SoC, hasil estimasi DNN dan hasil estimasi EKF. Dapat dilihat pada grafik tersebut bahwa hasil estimasi DNN telah mendekati data observasi dan mendapat hasil yang lebih baik dari metode pembanding yaitu EKF.

Dalam menentukan struktur DNN yang dapat mencapai akurasi yang baik, percobaan dilakukan beberapa kali dengan mengubah struktur *Deep Neural Network* (DNN) dan beberapa parameter *training* sampai didapatkan hasil yang diinginkan. Beberapa parameter *training* yang diubah antara lain adalah *Epochs, Learn Rate Drop Period, Initial Learn Rate, Learn Rate Drop Factor, Validation Frequency*, jumlah *Hidden Layer* dan Jumlah Hidden Neuron pada tiap *hidden layer*. Pada paper ini akan ditampilkan tiga percobaan dengan hasil terbaik.

Pada percobaan pertama, dilakukan dengan menggunakan 1000 Epochs, 1000 Learn Rate Drop Period, 0.01 Initial Learn Rate, 0.1 Learn Rate Drop Factor, 10 Validation Frequency. Untuk struktur DNN menggunakan 4 Hidden Layer dan 55 Hidden Neuron untuk tiap Hidden Layer.



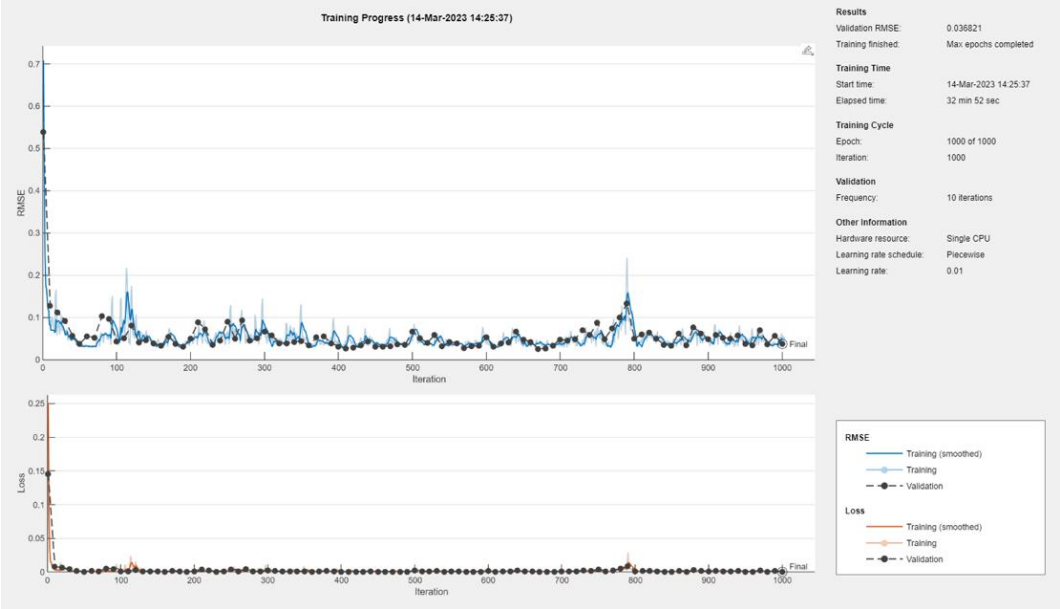
Gambar 4. *Training Plot* percobaan 1



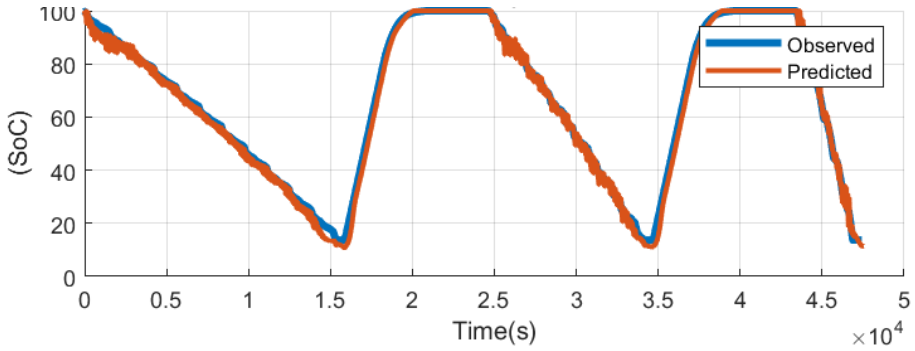
Gambar 5. Kurva perbandingan hasil estimasi dan nilai asli percobaan 1

Pada percobaan oertama ini, setealah dilakukan *training* dengan parameter *training* dan struktur DNN yang telah disebutkan, dievaluasi dengan Root Mean Square Error (RMSE) didapatkan hasil error sebesar 8.57%. Dapat dilihat pada *training plot* bahwa pada beberapa *epoch* terakhir *training* terdapat fluktuasi yang lumayan besar dan menyebabkan hasil yang didapatkan masih belum sesuai dengan yang diinginkan. Pada kurva perbandingan hasil estimasi dan data observasi juga dapat dilihat bahwa hasil estimasi masil belum begitu baik dengan adanya perbedaan yang lumayan besar dengan data observasi pada titik-titik tertentu.

Pada percobaan kedua, dilakukan dengan menggunakan parameter training yang sama seperti pada percobaan pertama. Perbedaan yang membedakan percobaan kedua ini adalah pada struktur DNN, yaitu dengan menambah jumlah *hidden layer* yang digunakan. Pada percobaan kedua dilakukan dengan menggunakan lima *hidden layer*.



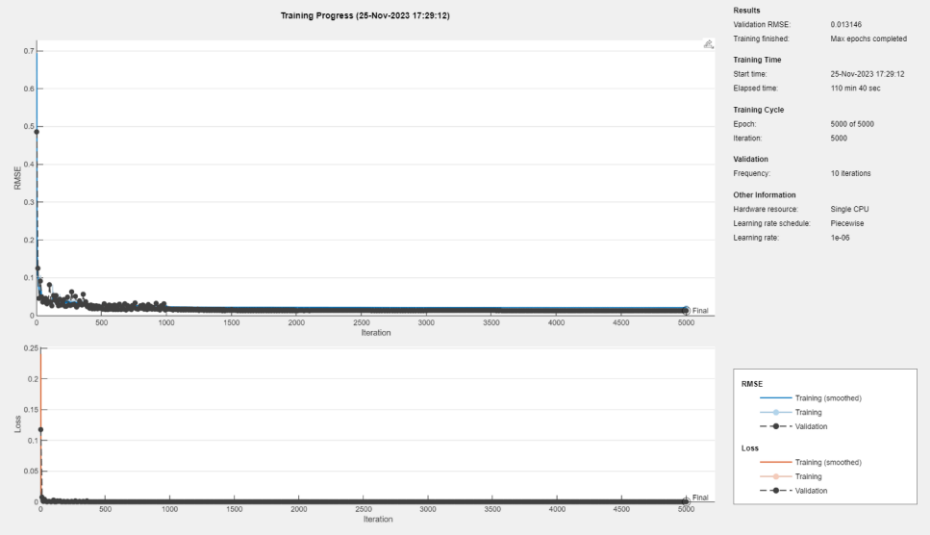
Gambar 6. *Training Plot* percobaan 2



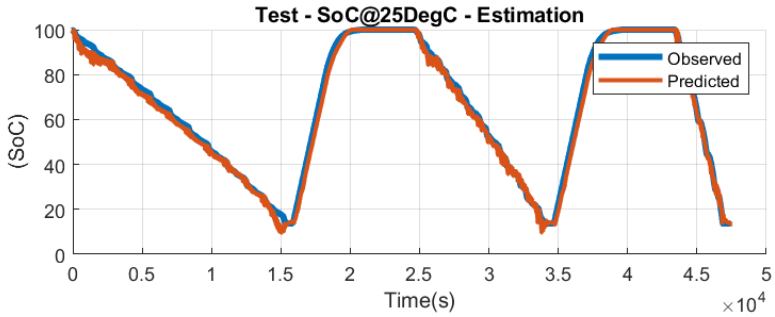
Gambar 7. Kurva perbandingan hasil estimasi dan nilai asli percobaan 2

Pada percobaan kedua ini, setelah dilakukan *training* dan evaluasi hasil *training* dengan RMSE, didapatkan nilai *error* sebesar 3.66%. Hasil *error* yang didapatkan kali ini sudah lebih baik dari *error* yang didapat pada percobaan pertama. Dapat dilihat pada *training plot*,pada saat proses *training* berjalan, masih belum baik dikarenakan selama proses tersebut masih menghasilkan hasil yang sangat fluktuatif. Hal ini menyebabkan hasil yang didapat dari *training* masih belum bisa dibilang konsisten. Pada kurva perbandingan data estimasi dan data observasi juga dapat dilihat pada siklus *discharge* yang kedua, estimasi yang dilakukan oleh DNN masih tersebar secara fluktuatif dan tidak menunjukkan konsistensi yang diinginkan.

Pada percobaan ketiga dilakukan beberapa perubahan pada parameter *training* yang dilakukan sehingga bisa mendapatkan hasil yang lebih baik. Perubahan yang dilakukan adalah menambah *epochs training* menjadi 5000 dan mengganti struktur DNN. Struktur DNN yang digunakan pada percobaan kali ini menggunakan enam *hidden layer*. Untuk jumlah *neuron* yang digunakan, pada *hidden layer* pertama menggunakan 60 *neuron*, *layer* kedua menggunakan 50 *neuron*, *layer* ketiga menggunakan 40 *neuron*, *layer* keempat menggunakan 30 *neuron*, *layer* kelima menggunakan 20 *neuron*, dan *layer* keenam menggunakan 10 *neuron.*



Gambar 8. *Training Plot* percobaan 3



Gambar 9. Kurva perbandingan hasil estimasi dan nilai asli percobaan 3

Dengan menggunakan parameter *training* dan struktur DNN yang demikian bisa dillihat pada *training plot* bahwa saat dilakukan *training*, pada prosesnya sudah tidak terjadi lagi fluktuasi hasil seperti yang terjadi pada percobaan sebelumnya. Fluktuasi dengan nilai yang tidak terlalu besar terjadi diawal proses *training*, setelah melewati 1000 *epoch* bisa dilihat bahwa hasil yang didapatkan sudah konsisten. Pada percobaan ketiga ini setelah dilakukan evaluasi *error* dengan RMSE didapat nilai *error* sebesar 1.31%. Nilai *error* tersebut sidah bisa dikatakan baik. Persebaran nilai estimasi yang dilakukan algoritma DNN juga lebih konsisten dan mendekati nilai observasi seperti yang terlihat pada kurva perbandingan data estimasi dan data observasi. Pada percobaan ketiga ini hanya terdapat *spike error* yang terjadi pada akhir siklus *discharge* pertama, selebihnya sudah mendekati data observasi dengan *error* sebesar 1.31%.

Percobaan dengan mengubah nilai parameter training, terutama jumlah epoch juga dilakukan. Tetapi dari percobaan yang telah dilakukan dengan nilai epochs melebihi 5000 sampai 10000 epoch, hasil yang didapat tidak berbeda jauh dengan yang didapatkan pada percobaan ketiga, dengan kompensasi waktu training yang dibutuhkan jauh lebih lama daripada percobaan ketiga. Mempertimbangkan hal tersebut, dari percobaan yang telah dilakukan ditetapkan bahwa percobaan ketiga adalah percobaan dengan hasil terbaik dan waktu training yang dibutuhkan adalah yang paling cepat.

Algoritma DNN tersebut diterapkan pada MATLAB Simulink, pada skema tersebut ditambahkan juga algoritma pembanding sebagai referensi. Untuk algoritma pembanding, algortima Extended Kalman Filter (EKF) digunakan. Dari hasil yang didapat dari skema Simulink tersebut, bisa dilihat pada Gambar 3. bahwa hasil estimasi yang dilakukan algoritma DNN mendapatkan nilai yang lebih baik daripada algoritma EKF.

SIMPULAN

Pada penelitian yang telah dilakukan, penulis mengajukan metode *Deep Neural Network* (DNN) digunakan sebagai metode untuk melakukan estimasi nilai *State-of-Charge* (SoC) pada baterai *Lithium-Ion* LG 18650 HG2. Hal tersebut dilakukan secara simulasi menggunakan MATLAB *Simulink*.pengujian dialkukan dengan membandingkan nilai SoC aktual pada baterai *Lithium-Ion* LG 18650 HG2 dengan nilai estimasi SoC yang dihasilkan dari hasil estimasi yang dilakukan algoritma DNN. Dari perbandingan tersebut nantinya akan didapatkan *error* estimasi nilai SoC. Dilihat dari hasil simulasi dari beberapa percobaan yang dilakukan untuk mendapat struktur DNN yang dapat menghasilkan hasil estimasi SoC paling mendekati nilai asli atau *real* SoC, pada percobaan pertama didapatkan nilai *error* sebesar 8.57% dengan struktur DNN; 4 *hidden layer*, 55 *neuron* pada tiap *hidden layer* dan 1000 *epochs* pada proses *training*. Pada percobaan kedua didapat *error* sebesar 3.66% dengan struktur DNN; 5 *hidden layer*, 55 *neurons* dan 1000 *epochs* pada proses *training*. Percobaan ketiga didapat nilai *error* 1.31% dengan struktur DNN; 6 *hidden layer*, 60 *neurons* pada *hidden layer* pertama, 50 *neurons* pada *hidden layer* kedua, 40 *neurons* pada *hidden layer* ketiga, 30 *neurons* pada *hidden layer* keempat, 20 *neurons* pada *hidden layer* kelima, 10 *neurons* pada *hidden layer* keenam dan 5000 *epochs* pada proses *training*. Dari ketiga percobaan tersebut didapatkan nilai *error* terkecil pada percobaan ketiga dengan nilai *error* 1.31% dari nilai asil SoC.

Hasil yang telah didapatkan dari pengujian secara simulasi ini didapat dengan *trial and error* dalam menentukan parameter *training*. Untuk mendapatkan parameter yang baik dan efisien kedepannya bisa ditambahkan algoritma pencarian yang dapat menentukan parameter tersebut secara otomatis. Pengujian *hardware* juga bisa dilakukan dan baterai LG 18650 HG2 yang nyata juga disarankan untuk dilakukan kedepannya, karena dengan demikian data yang yang didapatkan bisa lebih bervariasi.

DAFTAR RUJUKAN

International Energy Agency, I. (t.t.). World Energy Outlook 2018. www.iea.org/weo

Adebayo, T. S., Ullah, S., Kartal, M. T., Ali, K., Pata, U. K., & Ağa, M. (2023). Endorsing sustainable development in BRICS: The role of technological innovation, renewable energy consumption, and natural resources in limiting carbon emission. Science of The Total Environment, 859, 160181. https://doi.org/10.1016/j.scitotenv.2022.160181

Barman, P., Dutta, L., Bordoloi, S., Kalita, A., Buragohain, P., Bharali, S., & Azzopardi, B. (2023). Renewable energy integration with electric vehicle technology: A review of the existing smart charging approaches. Dalam Renewable and Sustainable Energy Reviews (Vol. 183). Elsevier Ltd. https://doi.org/10.1016/j.rser.2023.113518

Chemali, E., Kollmeyer, P. J., Preindl, M., & Emadi, A. (2018). State-of-charge estimation of Li-ion batteries using deep neural networks: A machine learning approach. Journal of Power Sources, 400, 242–255. https://doi.org/10.1016/j.jpowsour.2018.06.104

Hemeida, A. M., Hassan, S. A., Mohamed, A. A. A., Alkhalaf, S., Mahmoud, M. M., Senjyu, T., & El-Din, A. B. (2020). Nature-inspired algorithms for feed-forward neural network classifiers: A survey of one decade of research. Dalam Ain Shams Engineering Journal (Vol. 11, Nomor 3, hlm. 659–675). Ain Shams University. https://doi.org/10.1016/j.asej.2020.01.007

Hoeft, F. (2021). Internal combustion engine to electric vehicle retrofitting: Potential customer's needs, public perception and business model implications. Transportation Research Interdisciplinary Perspectives, 9. https://doi.org/10.1016/j.trip.2021.100330

Hossain Lipu, M. S., Hannan, M. A., Hussain, A., Ayob, A., Saad, M. H. M., Karim, T. F., & How, D. N. T. (2020). Data-driven state of charge estimation of lithium-ion batteries: Algorithms, implementation factors, limitations and future trends. Dalam Journal of Cleaner Production (Vol. 277). Elsevier Ltd. https://doi.org/10.1016/j.jclepro.2020.124110

Hossain Lipu, M. S., Hannan, M. A., Hussain, A., Saad, M. H., Ayob, A., & Uddin, M. N. (2019). Extreme learning machine model for state-of-charge estimation of lithium-ion battery using gravitational search algorithm. IEEE Transactions on Industry Applications, 55(4), 4225–4234. https://doi.org/10.1109/TIA.2019.2902532

Jiang, J., Chen, M., & Fan, J. A. (2021). Deep neural networks for the evaluation and design of photonic devices. Dalam Nature Reviews Materials (Vol. 6, Nomor 8, hlm. 679–700). Nature Research. https://doi.org/10.1038/s41578-020-00260-1

Lebrouhi, B. E., Khattari, Y., Lamrani, B., Maaroufi, M., Zeraouli, Y., & Kousksou, T. (2021). Key challenges for a large-scale development of battery electric vehicles: A comprehensive review. Journal of Energy Storage, 44, 103273. https://doi.org/10.1016/j.est.2021.103273

Ningrum, P., & Windarko, N. A. (2021). Estimation of State of Chrage (SoC) Using Modified Coulomb Counting Method with Open Circuit Compensation for Battery Management System (BMS). Dalam Journal on Advanced Research in Electrical Engineering (Vol. 5, Nomor 1).

Shu, X., Yang, W., Guo, Y., Wei, K., Qin, B., & Zhu, G. (2020). A reliability study of electric vehicle battery from the perspective of power supply system. Journal of Power Sources, 451. https://doi.org/10.1016/j.jpowsour.2020.227805

Vidal, C., Kollmeyer, P., Naguib, M., Malysz, P., Gross, O., & Emadi, A. (2020). Robust xEV Battery State-of-Charge Estimator Design Using a Feedforward Deep Neural Network. SAE Technical Papers, 2020-April(April). https://doi.org/10.4271/2020-01-1181

Xia, X., Li, P., Xia, Z., Wu, R., & Cheng, Y. (2022). Life cycle carbon footprint of electric vehicles in different countries: A review. Separation and Purification Technology, 301, 122063. https://doi.org/10.1016/j.seppur.2022.122063

Yang, F., Li, W., Li, C., & Miao, Q. (2019). State-of-charge estimation of lithium-ion batteries based on gated recurrent neural network. Energy, 175, 66–75. https://doi.org/10.1016/j.energy.2019.03.059

Yuen, B., Hoang, M. T., Dong, X., & Lu, T. (2021). Universal activation function for machine learning. *Scientific Reports*, *11*(1). https://doi.org/10.1038/s41598-021-96723-8

Zhang, M., & Fan, X. (2020). Review on the state of charge estimation methods for electric vehicle battery. Dalam *World Electric Vehicle Journal* (Vol. 11, Nomor 1). MDPI AG. https://doi.org/10.3390/WEVJ11010023.