Journal Name

ARTICLE TYPE

Cite this: DOI: 00.0000/xxxxxxxxxx

Performance of upcycled tyre-derived carbon in ethaline-glycol ionic electrolytes for sustainable supercapacitors: modelling cyclic voltammetry using standard and machine-learning methods.

Maria A. Sandoval-Riofrio,[1,a] and Kavisha Jayathunge[2,a]

Received Date
Accepted Date

DOI: 00.0000/xxxxxxxxxx

This study explores the development of sustainable supercapacitors using electrochemically activated carbon derived from plastic and tyre waste as electrode materials and ethaline, a deep eutectic solvent, as a green electrolyte. The electrochemical molten salts (EMS) process was employed to activate black carbon, a waste byproduct from tyre recycling, resulting in significant enhancements in carbon properties, including a 39.4% increase in BET surface area (from $43.1\,\mathrm{m}\cdot\mathrm{g}^{-2}$ to $60.09\,\mathrm{m}\cdot\mathrm{g}^{-2}$) and improved pore volume and size distribution. These findings establish the EMS process as a scalable and effective method for producing high-performance porous carbon materials. Supercapacitors assembled with EMS-activated carbon and ethaline exhibited remarkable electrochemical performance, achieving a specific capacitance of $210.56\,\mathrm{F}\cdot\mathrm{g}^{-1}$, surpassing many conventional aqueous and organic systems. Combined CV and EIS analyses revealed primarily capacitive behaviour with notable contributions from diffusion processes, particularly at elevated temperatures. Specific capacitances of 166.14, 121.63, and $69.81\,\mathrm{F}\cdot\mathrm{g}^{-1}$ at 45 °C, 65 °C, and 90 °C, respectively, underscored the thermal sensitivity and ionic mobility of ethaline. These results highlight the potential of integrating waste-derived carbon materials with environmentally friendly electrolytes to create efficient, sustainable energy storage systems. Further studies are recommended to optimise electrolyte performance and address decomposition mechanisms for broader applications.

1 Deep learning analysis

Machine learning models can be thought of as universal function approximators. That is, provided enough data to learn from, they are able to model any continuous, closed domain function to an arbitrary degree of accuracy[1]. Such models consist of interconnected "neurons" arranged in layers. These connections are dense, meaning each neuron is connected to all other neurons in the previous and next layer, giving rise to the common "fully connected" layer terminology. We will refer to a single layer as $F_i^{n,m}$ where i is the position of the layer, n is the input size (the number of input neurons expected), and m is the output size. The layer computes:

$$\hat{y} = F_i^{n,m}(x) = W_i x + b_i \tag{1}$$

where x is an n-dimensional vector, $\hat{y}$ is an m-dimensional vector, W_i is an $n \times m$ weight matrix and b_i is a constant term. Layers may be chained sequentially, as long as the inner dimensions of adjacent layers are equal, i.e. W_i and W_{i+1} can be multiplied. Such a sequence of connected layers is commonly referred to as a Multilayer Perceptron (MLP). An important element of the layers of the MLP is the activation function, which is applied to the output of each neuron, and introduces non-linearity into the model. This is essential for an MLP to fit the definition of a universal function approximator as defined earlier, because any sequence of fully connected layers must necessarily collapse into a single linear transform, which is not powerful enough to represent more complicated functions. The Rectified Linear Unit function (ReLU) is a popular activation function[2] designed for this purpose and is defined as follows:

$$\mathrm{ReLU}(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases} \tag{2}$$

Thus, the MLP must have a ReLU activation function between each hidden layer, giving the final model structure as laid out in Figure 1. Obtaining a $\hat{y}$ prediction from an MLP completes the "forward pass" part of training. The prediction is then evaluated against the target value y (also called the ground truth value) using a cost function L – see Equations 7 - 10 for how it is defined

[1] *Department of Engineering; E-mail: msandoval2@bournemouth.ac.uk*
[2] *National Centre for Computer Animation; E-mail: kjayathunge@bournemouth.ac.uk*
[a] *Talbot Campus, Bournemouth University, Fern Barrow, Poole BH12 5BB, UK*

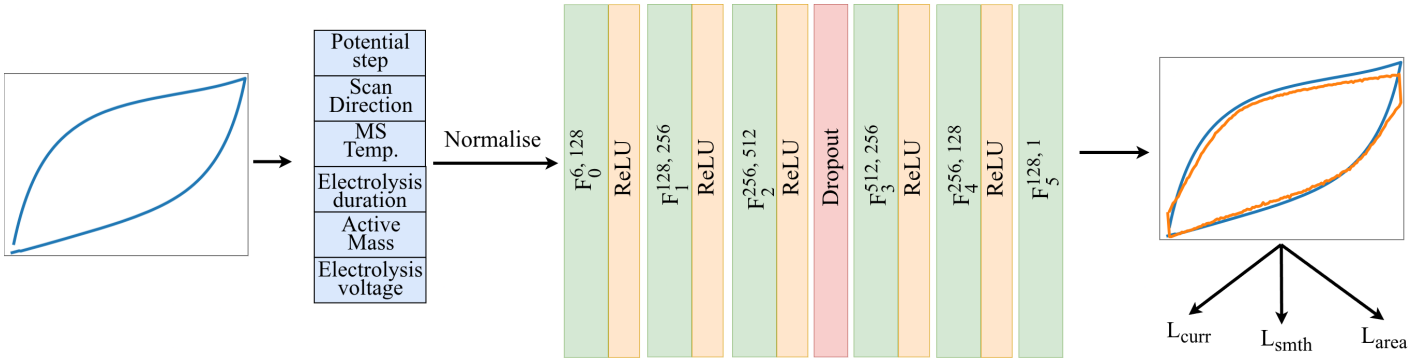

Fig. 1 An overview of the training pipeline and model architecture, incorporating dropout after the largest layer to discourage overfitting. The input is a 6-dimensional vector consisting of the potential step, and several experimental conditions under which the electrode material was sysnthesised, and the output is a prediction of the current value at that potential. Ground truth hysteresis curves shown in blue and model prediction shown in orange.

for this application. The partial derivative of L is calculated with respect to each parameter $w_i^{n,m}$ in W_i:

$$\frac{\partial L}{\partial w_i^{n,m}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w_i^{n,m}} \tag{3}$$

Computing all such partials completes the "backwards" pass of the training step – a process known as backpropagation – and gives us $\frac{\partial L}{\partial W_i}$, which tells the model how much, and the direction in which W should change in order to minimise L. Repeating these forward and backwards passes over the MLP over and over again brings the weights closer to their "ideal" values – ones for which L is minimised. In other words, the optimsing algorithm uses the difference between the output of the model and its target to update W in the correct direction. This is what is meant by training a machine learning model.

1.1 Previous work

The application of machine learning systems to the prediction of specific capacitance values is varied – some previous investigations involve predicting specific capacitance directly from experimental conditions[3], while others use these experimental conditions to predict hysteresis ***Note to Alex: Is this the correct terminology? Are CV curves hysteresis loops?*** curves[4]; the latter approach allows the indirect calculation of specific capacitance C_{sp}, which is proportional to the area between charge and discharge curves. It is this latter approach that is used by the present study, because predicting the full charge-discharge dynamics of cyclic voltammetry allows the model to generalise better to a wider range of conditions[5] – especially important given the limited data available for this study. Using this approach, C_{sp} is given by:

$$C_{sp} = \frac{\int I_{pred}(E)\,dE}{2 \cdot \nu \cdot m \cdot \Delta V} \tag{4}$$

where I_{pred} is the function that is approximated by the MLP, E is the potential, ν is the scanrate, m is the total mass of the active material in the electrodes, and ΔV is the change in voltage over the whole charge/discharge cycle.

Table 1 Characteristics of each cycle that constitutes the dataset used for training our model. The scanrate was $100\,\mathrm{mV \cdot s^{-1}}$ in all cases.

Cycle	M / mg	T / °C	V_e / V	D / h	Electrolyte
BCMS2	0.11	800	2	2	Ethaline
BCMS3	0.11	750	2	2	Ethaline
BCMS5	0.18	650	2	2	Ethaline
BCMS7	0.25	700	2.8	2	Ethaline
BCMS9	0.25	750	2.8	2	Ethaline

1.2 Dataset

The complete dataset is made up of current density samples ***(ask Alex if this is the correct term)*** from 5 cyclic voltammetry cycles – see Table 1. These were collected via potentiostat with a scanrate of $100\,\mathrm{mV \cdot s^{-1}}$. Each cycle consists of 264 rows, with columns for current (I / A), voltage step (E / V), active electrode mass (M / mg), temperature (T / °C), electrolysis voltage (V_e / V), electrolysis duration (D / h), and scan direction (S). The last feature tells the model if the particular point is on the charge or discharge curve and is represented by either +1 (charging) or -1 (discharging).

Notice that the various inputs span many orders of magnitude – this is a problem for machine learning models, which work best when their inputs are all near or around -1 to 1 (rephrase this). Using raw features such as those presented in Table 1 mean that ones with large absolute values dominate the weights of the model, even though they may not be as important, and vice versa, which can lead to lower performance[6]. For this reason, all features are scaled across all cycles such that they have zero mean and unit standard deviation. To transform an individual instance of a feature d_i (e.g. electrode mass, temperature) to its scaled value $d_{norm,i}$:

$$d_{norm,i} = \frac{d_i - \mu(D)}{\sigma(D)} \tag{5}$$

where D is the collection of all the values for this feature, μ is the mean of values in D, and σ is their standard deviation. These scaled values are used in all subsequent machine learning operations, and final outputs are de-normalised using the inverse of Equation 5 for display and comparison with experimental values.

Current values were also transformed in this way. However, the potential step E was rescaled slightly differently: rather than ensuring zero mean and unit variance, we rescale them such that the minimum potential across the whole dataset is -1 and the maximum is +1:

$$E_{mid} = \frac{E_{max} + E_{min}}{2}$$
$$E_{half-range} = \frac{E_{max} - E_{min}}{2}$$
$$E_{norm} = \frac{E - E_{mid}}{E_{half-range}} \quad (6)$$

where E_{min} and E_{max} are the minimum and maximum potential values *across the whole dataset*. This scaling ensures that all cycles share a common and bounded potential domain, making it easier for the model to lear consistent relationships across different experimental conditions.

The input to the model is therefore a 6 dimensional, normalised feature vector that consists of a potential step and experimental conditions under which the electrode material was synthesised, and the output is the current value at the potential step.

1.3 Experimental design and model architecture

We employ leave-one-out cross validation when training: for any one run, the model is trained only on points from 4 of the cycles; points from the 5^{th} are held out for validation. Here, validation means only evaluating the forward pass of the model and not updating the weights of the model. It is a test to determine if the model is able to generalise to "unseen" data, and each cycle is subjected to this treatment in turn. We repeat training runs five times independently to build confidence that the model's output is reliable.

Our model is a multilayered perceptron with 4 hidden layers in between a 6-neuron input layer and a 1-neuron output layer. Please refer to Figure 1 for the full model architecture. Note the dropout layer, which randomly (with some probability p) deactivates a subset of neurons during training. This is done so that the model does not rely too heavily on any single neuron or set of neurons, thereby reducing overfitting and improving generalisation to new data[7]. Wider layers with more neurons benefit more from dropout as they are more at risk from overfitting[8], hence the placement between F_2 and F_3.

As mentioned in the Section 1, the objective function scores the output of the MLP so it can be used in gradient calculations, which in turn are used to inform updates to the model's weights. Because the model predicts the current at a particular voltage step, part of the loss function is the mean squared error (MSE) between the predicted and actual value. This is given by:

$$L_{curr} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \quad (7)$$

where $\hat{y}_i$ is the predicted current value, y_i is the true value, and N is the number of predicted points. Further, in order to maintain a smooth curve that is physically consistent with the behaviour of real hysteresis curves, we impose a penalty on the second derivative of the predicted current with respect to potential, discouraging sharp local curvature and jitter:

$$L_{smth} = \left\| \frac{\partial^2 I_{pred}}{\partial E^2} \right\|_2^2 \quad (8)$$

where I_{pred} is the predicted current and E is the potential. Finally, we also use the area of the predicted curve in another MSE loss calculation to encourage the model to produce the correct net area exhibited by the experimental curve:

$$L_{area} = \frac{1}{M}\sum_{j=1}^{M}(\hat{a}_j - a_j)^2 \quad (9)$$

where M is the number of cycles, a_j is the actual area between the curves for a given cycle, and $\hat{a}_j$ is the area calculated using predicted current values. The losses are combined for final objective for the MLP:

$$L = L_{curr} + L_{smth} + L_{area} \quad (10)$$

In order to assess the importance of these loss functions, an ablation study was conducted to measure the impact of each objective on the final prediction. A comparison of C_{sp} predicted by models under these experimental settings is given in Figure 2, and a comparison of their relative errors is given in Table 2. The base MLP, henceforth called "Base" is only tasked with the L_{curr} objective. The following experimental modifications were added to the Base model:

1. "+ D" – inclusion of a **D**ropout layer, as discussed in Section 1.3
2. "+ S" – addition of the **S**moothing L_{smth} objective to L
3. "+ A" – addition of the **A**rea L_{area} objective to L

Table 2 Average error (relative %) for each cycle over 5 independent runs. Base+D+A exhibits the least overall error across all the settings.

Cycle	Base	Base +D	Base +D +S	Base +D +S +A	Base +D +A
BCMS2	0.62	0.58	2.54	7.17	0.77
BCMS3	29.82	23.28	20.15	16.11	16.93
BCMS5	7.04	15.05	7.88	2.61	8.83
BCMS7	30.87	25.37	18.50	26.78	17.92
BCMS9	9.12	4.64	1.91	2.42	0.39
Avg. Err(%)	15.49	13.79	10.20	11.02	**8.97**

1.4 Results and discussion

Results were collected for each experimental setting described in Section 1.3. Figure 2 reports the averaged, normalised C_{sp} values computed over five independent runs. Predicted C_{sp} were normalised by their corresponding ground-truth values to facilitate comparison across settings, as C_{sp} spans several orders of magnitude. Error bars indicate the 95% confidence interval over the five runs.

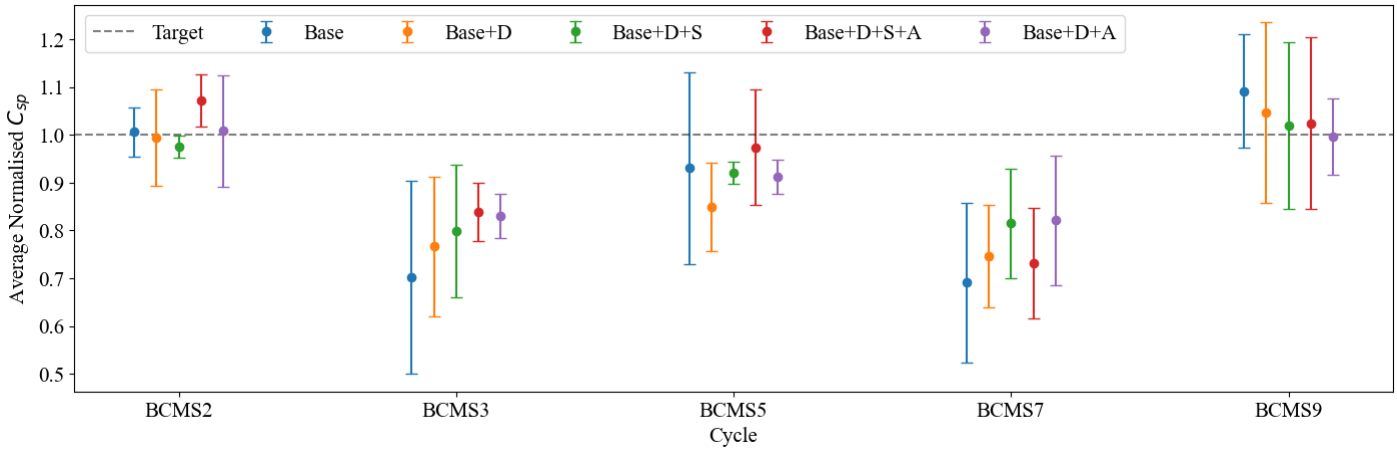

Fig. 2 A plot reporting the average normalised C_{sp} over 5 runs per cycle per experimental setting. Error bars indicate 95% confidence intervals over runs.

Overall, the Base+D+A configuration (corresponding to the model incorporating both dropout and the area loss of Equation 9) exhibits the samllest deviation from the C_{sp} baseline. This setting also yields comparatively narrower confidence intervals, suggesting improved stability across runs. Incorporating the area-based loss term (Equation 9) resulted in improved performance across all curves. As shown in Figure 2, models that include this loss term tend to produce predictions that are closer to the target baseline. It should be noted that the smoothness loss term (Equation 8) seems to be somewhat at odds with the area loss. This effect is reflected in Table 2, which presents the relative error % between the predicted and true C_{sp} values across all the experimental settings. The addition of area loss to Base+D+S results in a slight increase the average error. However, removing the smoothness loss (corresponding to the Base+D+A configuration) yields the lowest overall error. This suggests that the two loss terms may impose competing constraints on the learned representations, which could partially limit their combined effectiveness.

A key limitation of the experimental setup lies in the limited diversity of the input features. Although each of the five cycles contains 264 samples, all samples within a given cycle share the same experimental conditions; the only varying factor is the voltage step. Moreover, the voltage progression itself is identical across all cycles. Consequently, when accounting for both forward and backward curves, the dataset comprises only around 264 unique feature vectors across 1320 samples. This extremely low feature variance substantially constrains the expressive capacity available to the model and limits the generalisability of the learned representations. In this context, the fact that the model exhibits meaningful performance at all is somewhat surprising, and underscores the need for more diverse experimental conditions (i.e., a wider range of these conditions) in future studies.

Conclusions

The conclusions section should come in this section at the end of the article, before the Author contributions statement and/or Conflicts of interest statement.

Author contributions

We strongly encourage authors to include author contributions and recommend using CRediT for standardised contribution descriptions. Please refer to our general author guidelines for more information about authorship.

Conflicts of interest

In accordance with our policy on Conflicts of interest please ensure that a conflicts of interest statement is included in your manuscript here. Please note that this statement is required for all submitted manuscripts. If no conflicts exist, please state that "There are no conflicts to declare".

Data availability

A data availability statement (DAS) is required to be submitted alongside all articles. Please read our full guidance on data availability statements for more details and examples of suitable statements you can use.

Acknowledgements

The acknowledgements come at the end of an article after the conclusions and before the notes and references.

Notes and references

1 P. Geuchen, T. Jahn and H. Matt, *Constructive Approximation*, 2025, **62**, 361–402.

2 A. F. Agarap, *Deep Learning Using Rectified Linear Units (ReLU)*, 2019.

3 G. Kumar Yogesh, D. Nandi, R. Yeetsorn, W. Wanchan, C. Devi, R. Pratap Singh, A. Vasistha, M. Kumar, P. Koinkar and K. Yadav, *Energy Advances*, 2025, **4**, 119–139.

4 A. Ravichandran, V. Raman, Y. Selvaraj, P. Mohanraj and H. Kuzhandaivel, *ACS Omega*, 2024, **9**, 33459–33470.

5 S. Deebansok, J. Deng, E. Le Calvez, Y. Zhu, O. Crosnier, T. Brousse and O. Fontaine, *Nature Communications*, 2024, **15**, 1133.

6 Y.-S. Kim, M. K. Kim, N. Fu, J. Liu, J. Wang and J. Srebric, *Sustainable Cities and Society*, 2025, **118**, 105570.
7 W. Mou, Y. Zhou, J. Gao and L. Wang, Proceedings of the 35th International Conference on Machine Learning, 2018, pp. 3645–3653.
8 N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, *Journal of Machine Learning Research*, 2014, **15**, 1929–1958.