Unsupervised Classification of Voltammetric Data beyond Principal Component Analysis

Christopher Weaver,[a] Adrian Fortuin,[a,b] Anton Vladyka[a] and Tim Albrecht*[a]

[a] School of Chemistry, University of Birmingham, Edgbaston Campus, Birmingham B15 2TT, United Kingdom

[b] Faculty of Mechanical Engineering, Helmut Schmidt University, 22043 Hamburg, Germany

t.albrecht@bham.ac.uk

Supporting Information

1) Simulation of electrochemical data

All input cyclic voltammograms were simulated in DigiSim® v3.0. Temperature, scan rate, uncompensated resistance and double layer capacitance were kept constant at 298.15 K, 50 mV s^{-1}, 10 Ω, and 20 μF cm^{-2} respectively, for all reactions. Electrode radii were varied from 10^{-6} cm to 10^{-1} cm in increments of one order of magnitude. Kinetic data was based on the electrochemical oxidation of aniline to polyaniline and is summarized in Table 1.[1,2]

Table 1: Thermodynamic and kinetic parameters for reaction modelling

Variable / Unit	Reaction 1	Reaction 2	Reaction 3
Diffusivity / cm^2 s^{-1}	9.2×10^{-6}	9.2×10^{-6}	4.6×10^{-6}
Redox Potential / V	1.294		0.654
Electron transfer rate / cm s^{-1}	3×10^{-2}	-	10
Transfer coefficient	0.65	-	0.50
Initial concentration / mol L^{-1}	0.05	0.00	0.00
Reaction rate constant / s^{-1}	-	1×10^{4}	-
Reaction equilibrium constant / -	-	250	-
Mechanism	A + e = B	B + A = C	C + 2e = P

2) Data image processing

CVs were first plotted on current vs potential axes. The current values for each plot were normalised such that values were scaled between -1and 1. This made up a 600-dimensional dataset of data values. Each plot was then converted to an RGB image of size 1080 x 1080 x 3 which were normalised by dividing by the maximum pixel value, 255. These images were then both flattened into a raw 3499200-dimensional dataset, and put through a feature extractor followed by flattening into a features 557568-dimensional dataset. The feature extraction process utilised a topless VGG-16 CNN to compress the raw traces into feature traces.

Once the datasets were constructed they were passed through the three dimensionality reduction algorithms: PCA, t-SNE, and UMAP. The hyperparameters chosen for each technique are summarised in Table 2 and Table 3. All parameters not specified were left as their default values.

This analysis process was implemented in Python v3.8.11. The VGG-16 network, architecture and weights, were provided by Tensorflow v2.3.0. PCA and t-SNE algorithms were provided by scikit-learn v0.24.2. Lastly, UMAP was provided by umap-learn v0.5.1.

3) Hyperparameter Optimisation

3.1) t-SNE Parameters

To find the optimum parameters for both silhouette score and perimeter score for each of the three datasets, a 2D grid search was performed on the two main t-SNE hyperparameters. These are the perplexity, and learning rate parameters. Once completed, the search produced a 2D matrix for both silhouette and perimeter scores on raw data, images data, and features data. The elements of each matrix were removed if the corresponding KL-divergence was greater than 0.6 then the argmax of each matrix was calculated to find the best parameters.

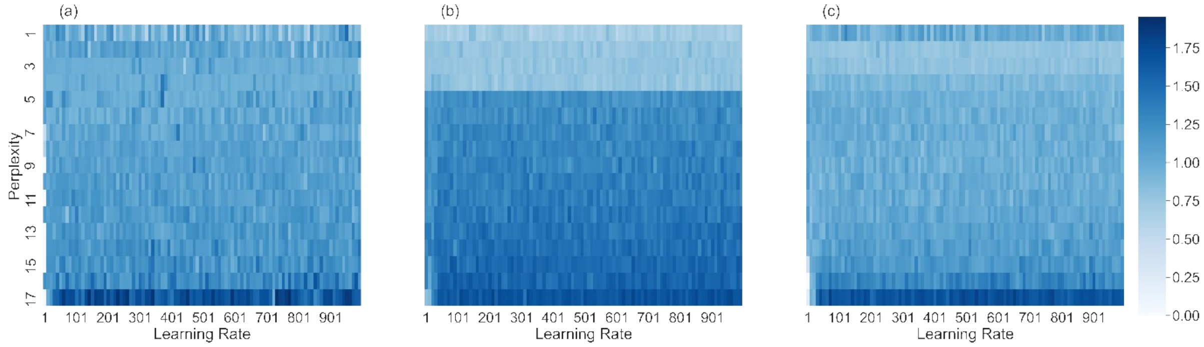


FIG S1 Heatmaps showing the average perimeter score for each combination of learning rate and perplexity. Results are shown for the raw data dataset (a), images dataset (b), and the features dataset(c).

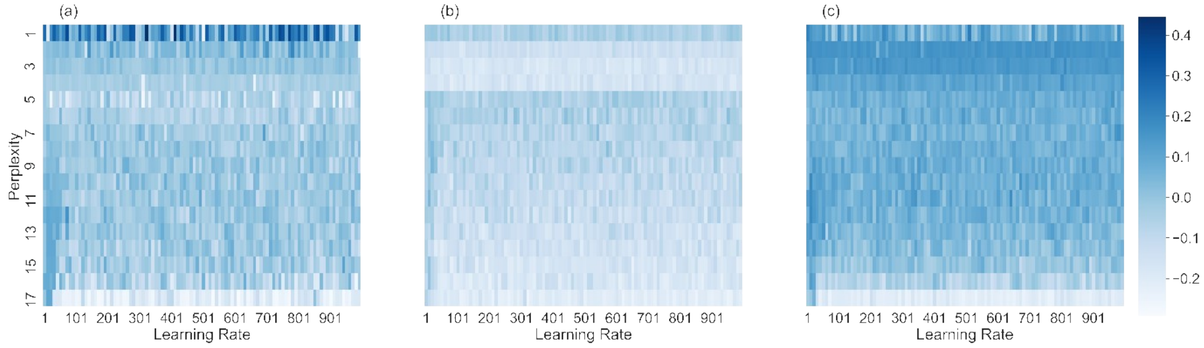


FIG S2 Heatmaps showing the average silhouette score for each combination of learning rate and perplexity. Results are shown for the raw data dataset (a), images dataset (b), and the features dataset(c).

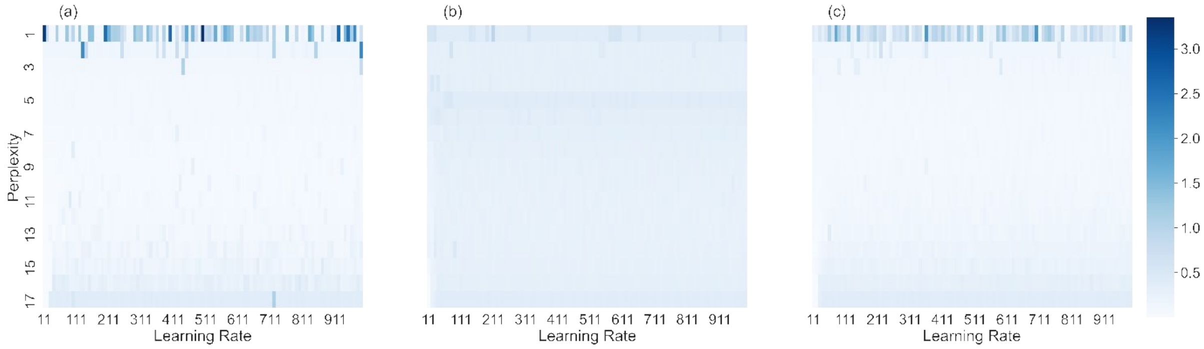


FIG S3 Heatmaps showing the average KL-Divergence loss after training for each combination of learning rate and perplexity. Results are shown for the raw data dataset (a), images dataset (b), and the features dataset(c).

Table 2 Hyperparameters chosen for t-SNE for maximising both perimeter and silhouette scores alongside their KL-Divergence losses after training.

Data	**Perplexity**	**Learning Rate**	**KL-Divergence**
Perimeter Score	17	771	0.38
Silhouette Score	1	411	0.38

Images	**Perplexity**	**Learning Rate**	**KL-Divergence**
Perimeter Score	17	211	0.38
Silhouette Score	9	21	0.38

Features	**Perplexity**	**Learning Rate**	**KL-Divergence**
Perimeter Score	17	71	0.38
Silhouette Score	1	511	0.38

(NB: Each t-SNE run utilised PCA initialisation provided by scikit-learn and was limited to 5000 iterations).

3.2) UMAP Parameters

To find the optimum parameters for both silhouette score and perimeter score, a 3D grid search was performed on the three main UMAP hyperparameters. These are the neighbourhood size (n_neighbors), minimum distance (min_dist), and learning rate parameters. Once completed, the search produced a 3D matrix for both silhouette and perimeter scores on raw data, images data, and features data. During the search, some of the runs produced a warning that the graph produced by spectral embedding was disconnected. These corresponding runs were removed. Then the argmax of each matrix was calculated to find the best parameters.

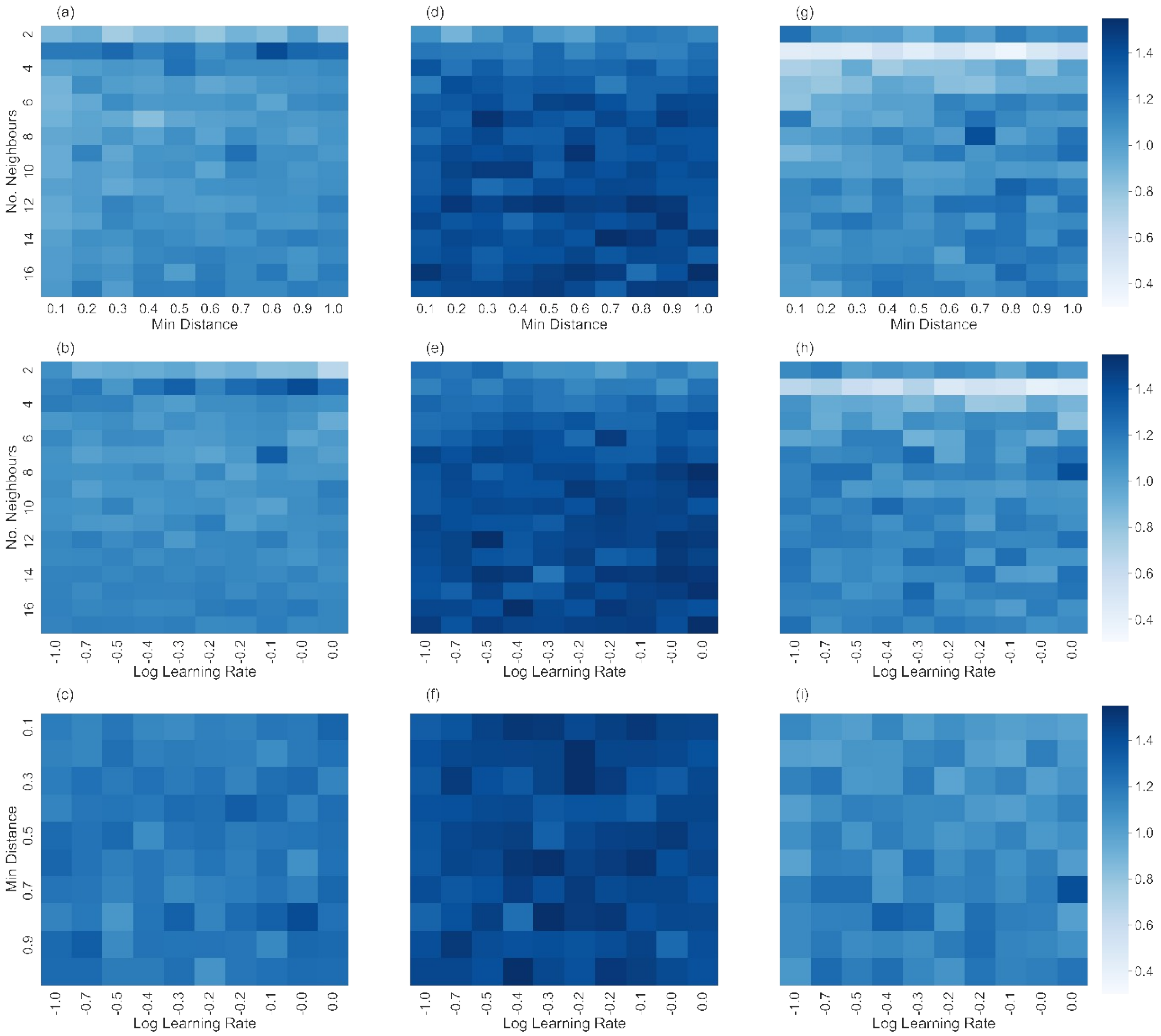


FIG S4 *Three slices of the 3D grid search matrix for perimeter score centred around the maximum score. Slices are shown for both the raw data dataset (a-c), the images dataset (d-f) and the features dataset(g-i).*

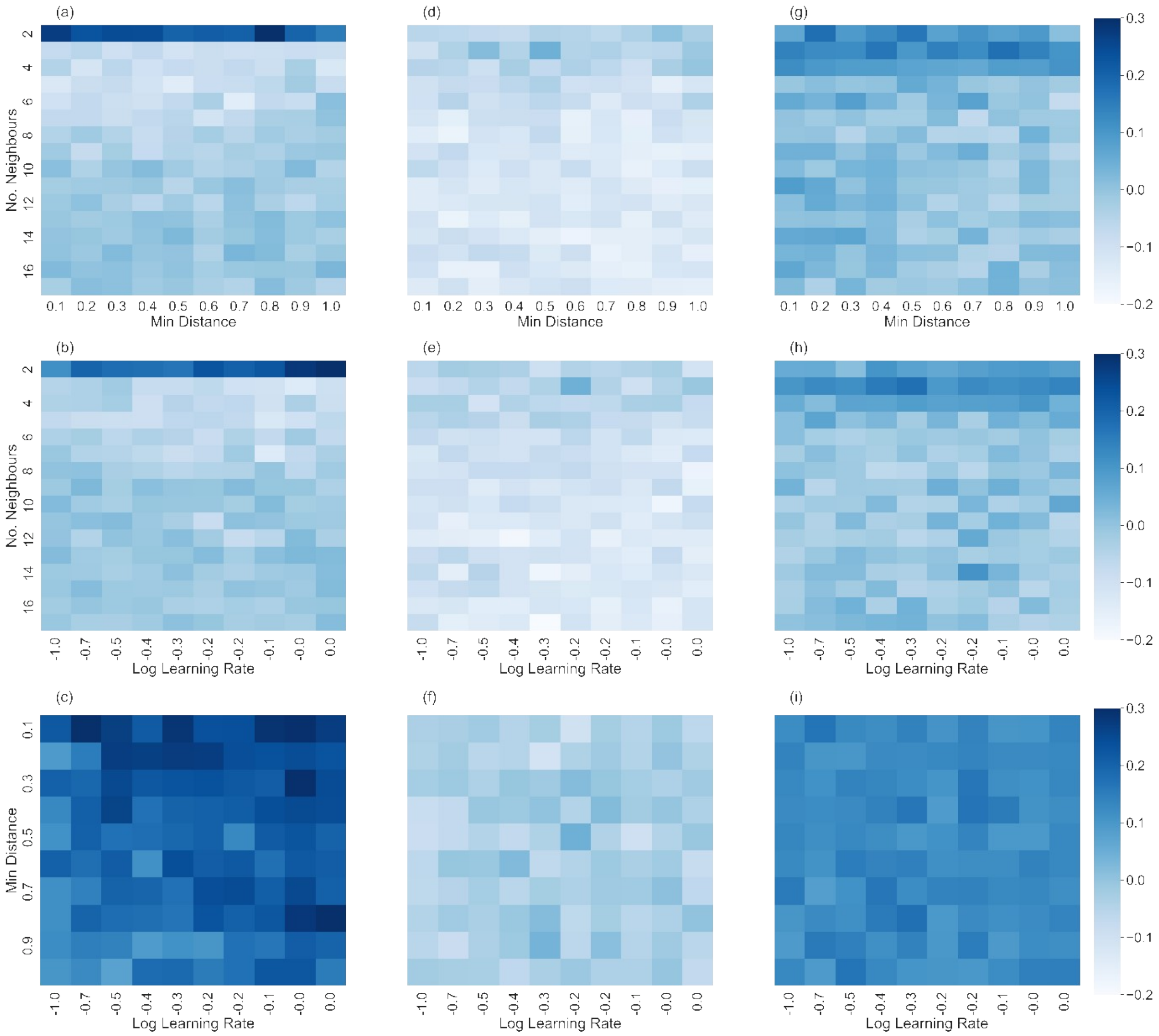


FIG S5 Three slices of the 3D grid search matrix for silhouette score centred around the maximum score. Slices are shown for both the raw data dataset (a-c), the images dataset (d-f), and the features dataset(g-i).

Table 3 Hyperparameters chosen for UMAP for maximising both perimeter and silhouette scores.

Data	**No. Neighbours**	**Min Distance**	**Learning Rate**
Perimeter Score	3	0.8	0.9
Silhouette Score	2	0.8	1.0

Images	**No. Neighbours**	**Min Distance**	**Learning Rate**
Perimeter Score	16	1.0	0.4
Silhouette Score	3	0.5	0.6

Features	**No. Neighbours**	**Min Distance**	**Learning Rate**
Perimeter Score	8	0.7	1.0
Silhouette Score	3	0.8	0.5

4) CVs and filter outputs

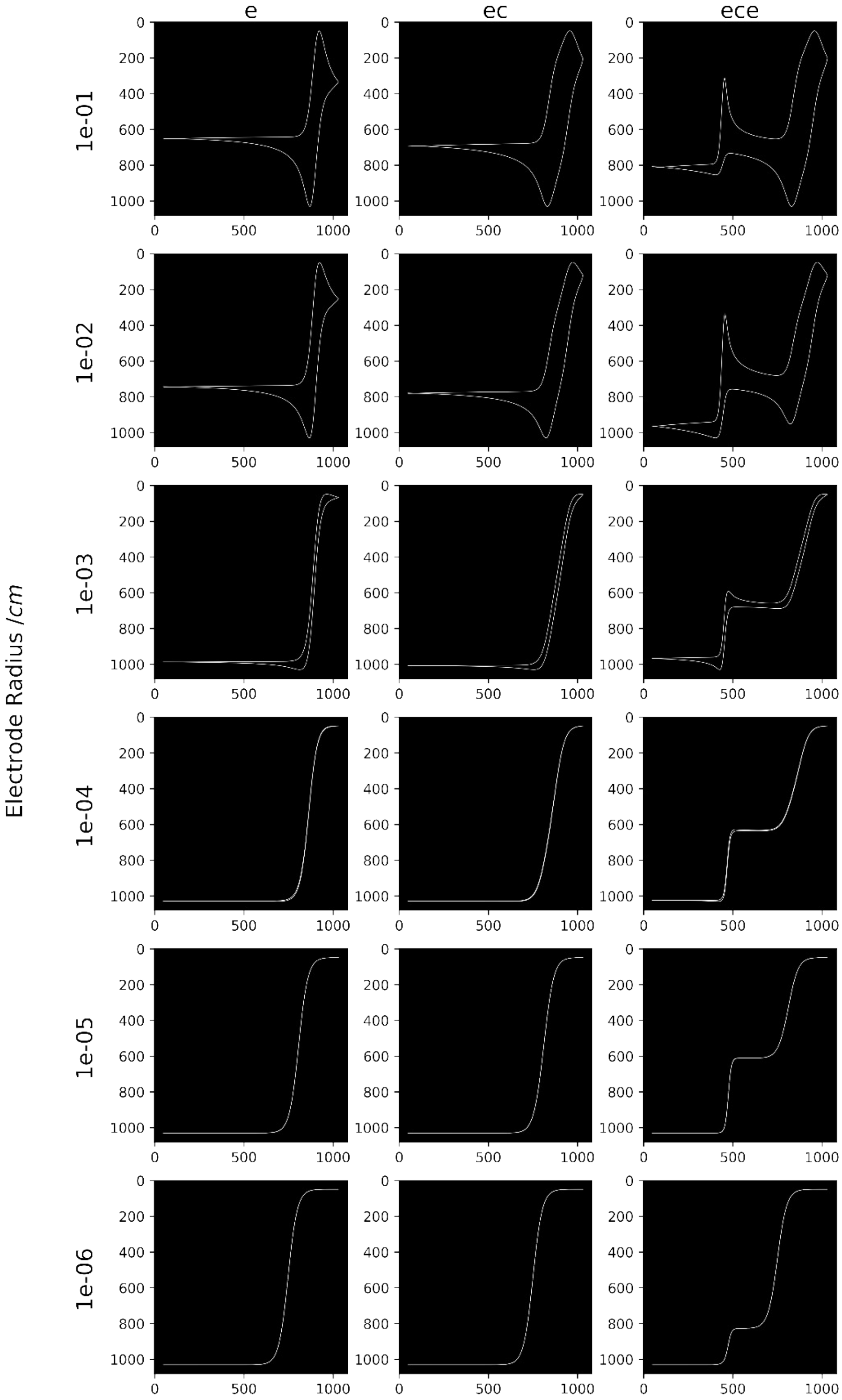


FIG S6 Raw images of all CVs generated for the 3 mechanisms at 6 different electrode radii. Each image corresponds to a 1080 x 1080 x 3 RGB matrix.

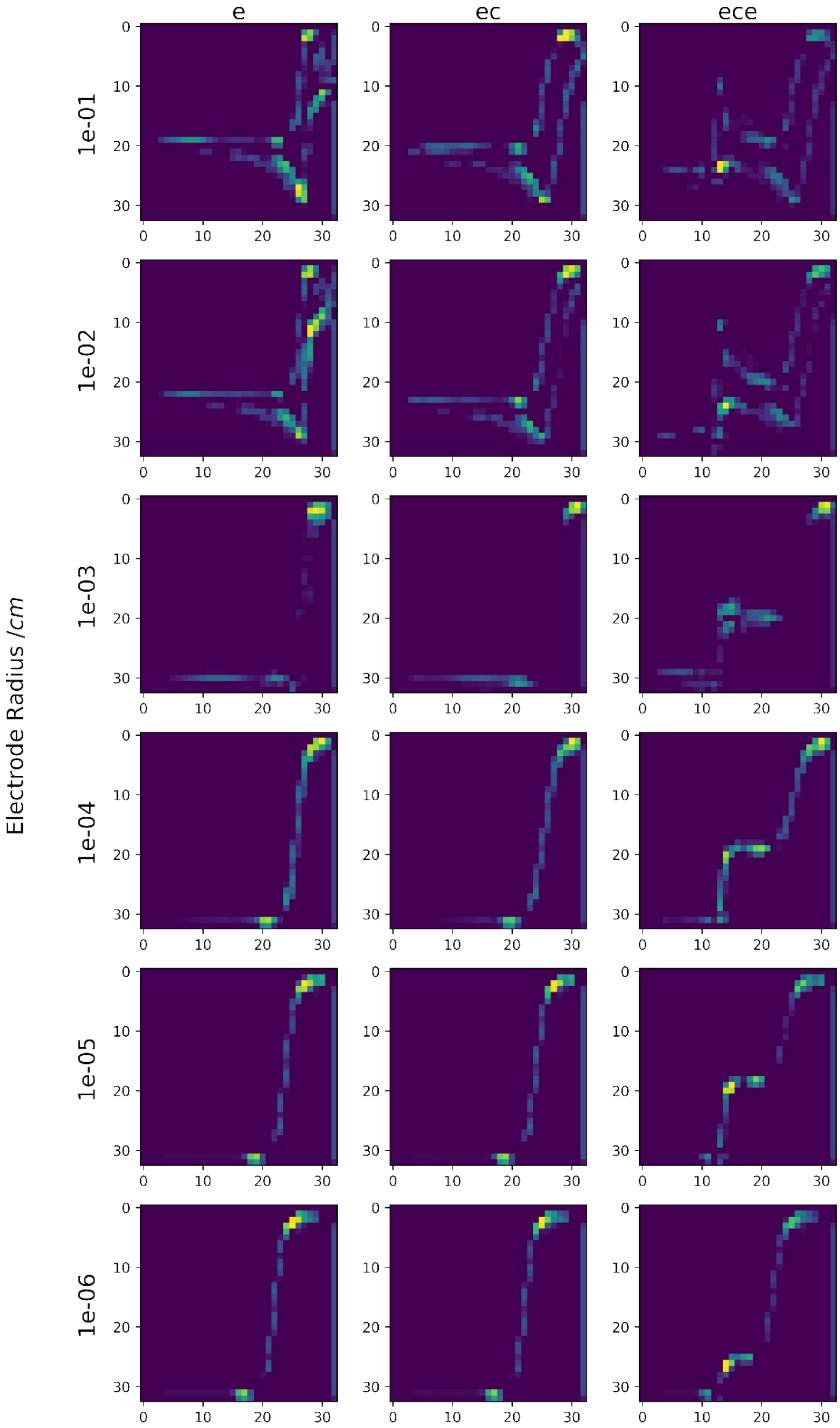
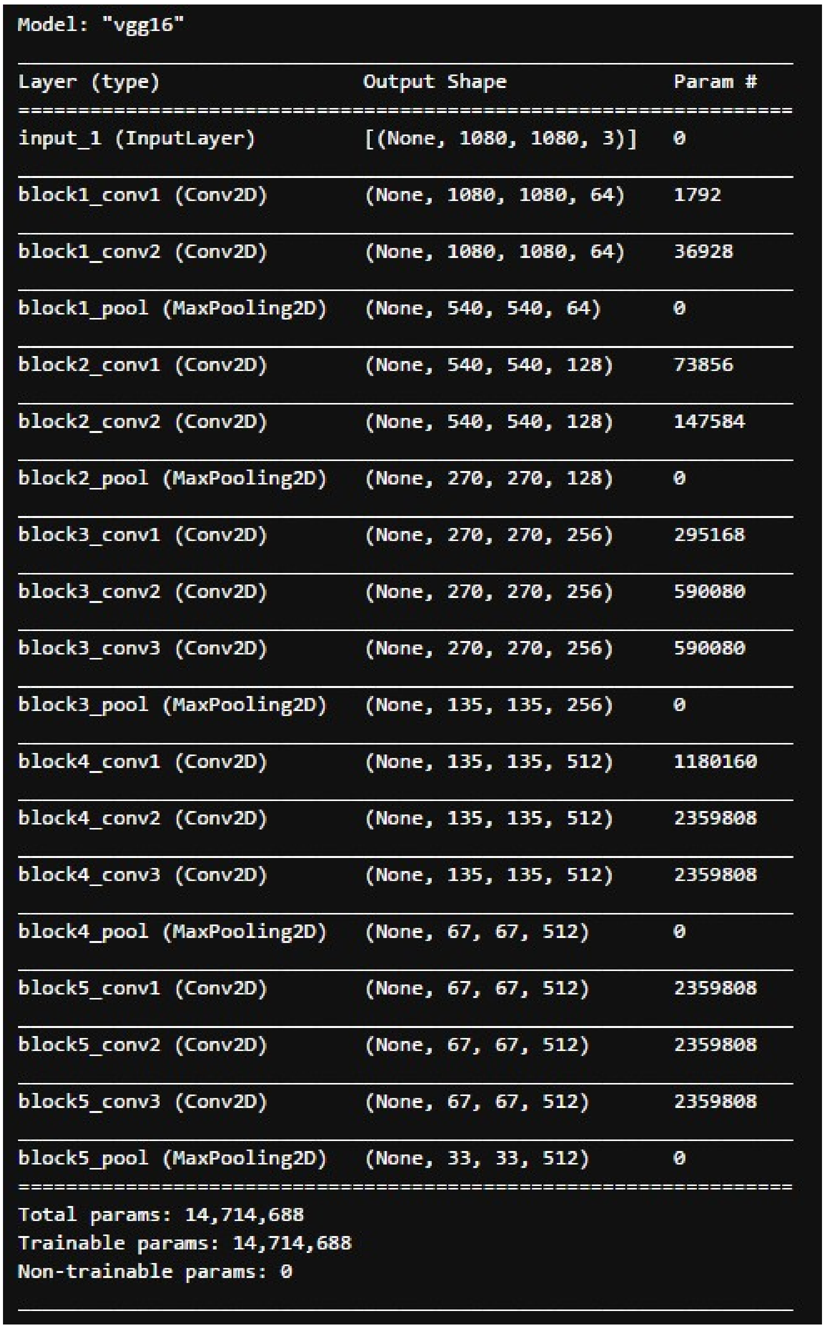


FIG S7 Example slice of the 33 x 33 x 512 matrices outputted by the convolutional head of the VGG-16 neural network for each CV.



```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 1080, 1080, 3)]   0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 1080, 1080, 64)    1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 1080, 1080, 64)    36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 540, 540, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 540, 540, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 540, 540, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 270, 270, 128)     0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 270, 270, 256)     295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 270, 270, 256)     590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 270, 270, 256)     590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 135, 135, 256)     0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 135, 135, 512)     1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 135, 135, 512)     2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 135, 135, 512)     2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 67, 67, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 67, 67, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 67, 67, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 67, 67, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 33, 33, 512)       0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

FIG S8 Architecture of the VGG-16 convolutional head used for generating the features dataset.

5) Additional Results

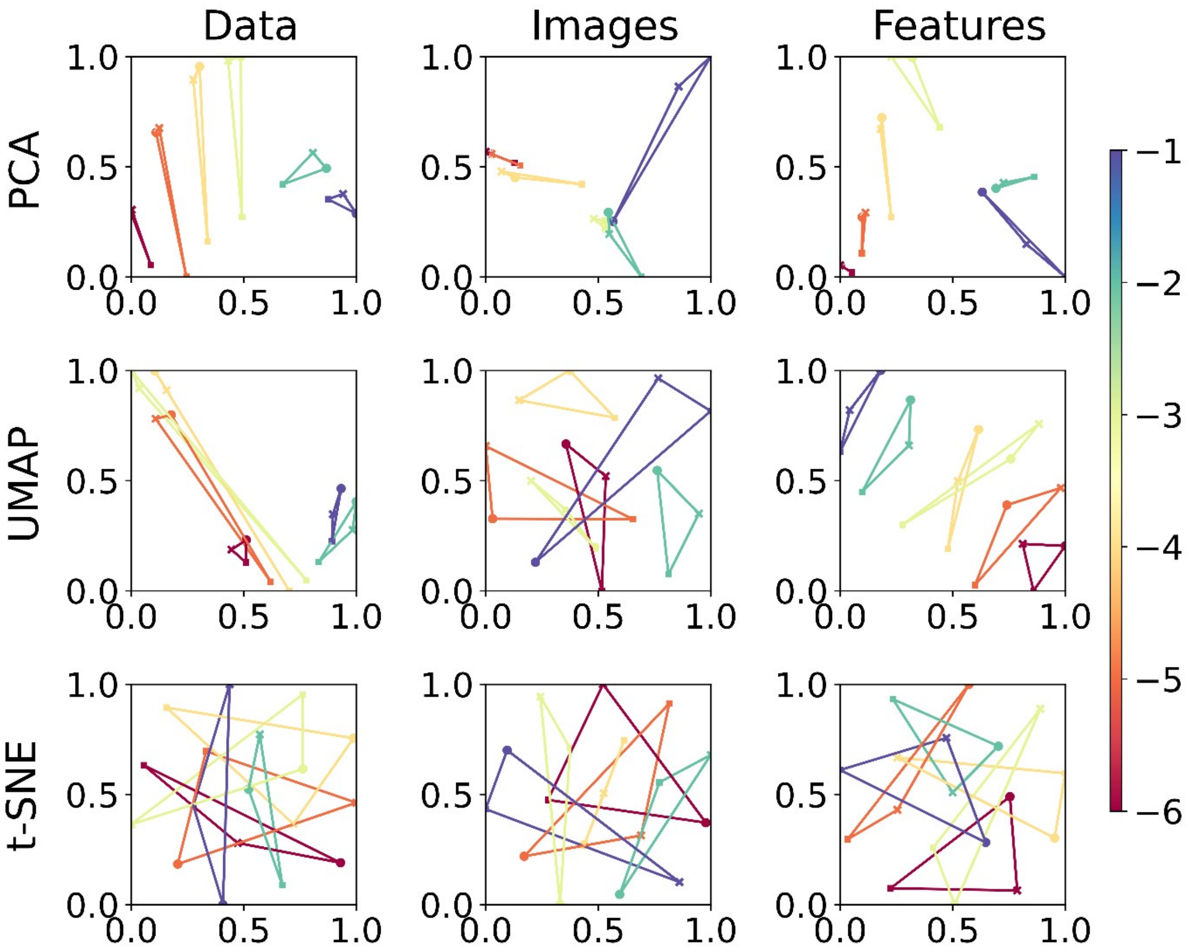


FIG S9 Scatter plots showing one example repeat over all electrode radii for each combination of dataset and dimensionality reducer when parameters optimised of maximum perimeter score were utilised.

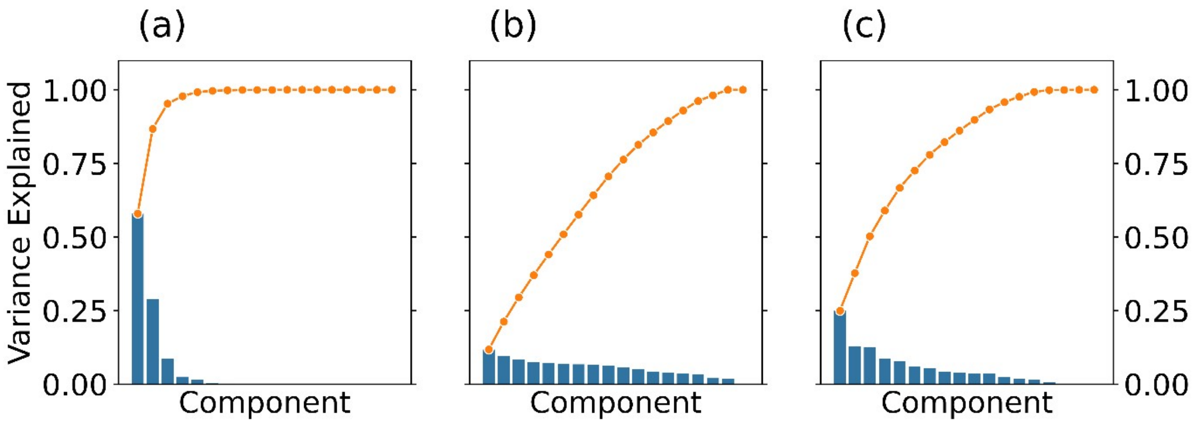


FIG S10 Scree plots showing the explained variance for each principal component (blue) and their cumulative quantities (orange) when applied to the raw data (a), images (b), and features (c) datasets.

6) The effect of noise on the classification performance

In order to investigate the robustness of the classification in the presence of electric noise that would typically be observed in a CV experiment, a noise component of constant magnitude was added to simulated data (independent of potential, normally distributed). Its standard deviation σ was taken relative to the scaled maximum current range from -1 to +1, as described in the main text. The figure below shows the corresponding image representation (1000 by 1000 pixels) for the three mechanisms and six electrode radii, for s = 2.5%.

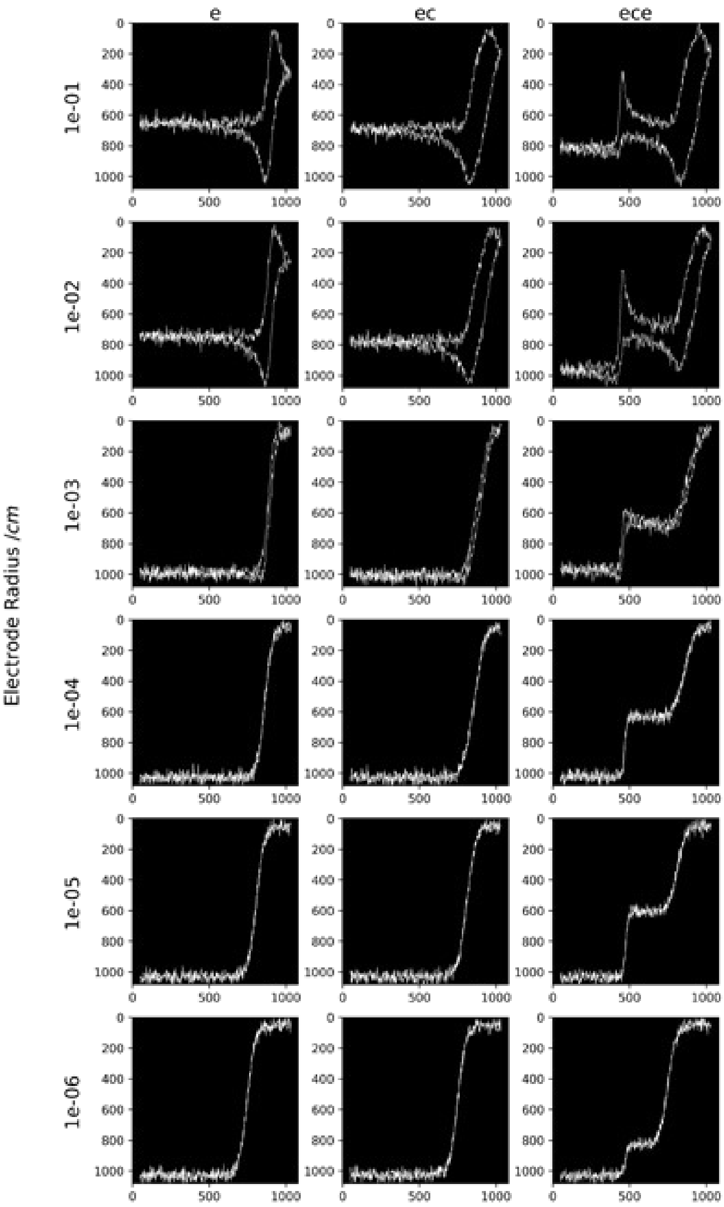


FIG S11 Simulated CV data in the presence of electric noise of constant magnitude, σ = 2.5%. Another set was generated for σ = 1.25% (not shown).

Generally, the presence of noise would be expected to increase overlap between similar CVs and hence render the differentiation more difficult (either between different mechanisms or electrode radii). Indeed, for the perimeter score *P* (using accordingly optimised hyperparameters), this simple expectation appears to hold true, left column in fig. S12 below: the sequence of overall performance remains the same (t-SNE > UMAP > PCA), even though the numerical values seem to decrease with an increasing amount of noise present.

As for the noise-free data, the picture is however more complex for the *S* score, i.e. when the choice of hyperparameters is adapted. For PCA applied to the raw numerical data, the performance remains rather constant, application to images leads to the poorest performance of the three, while the performance when applied to features decreases somewhat, albeit not dramatically.

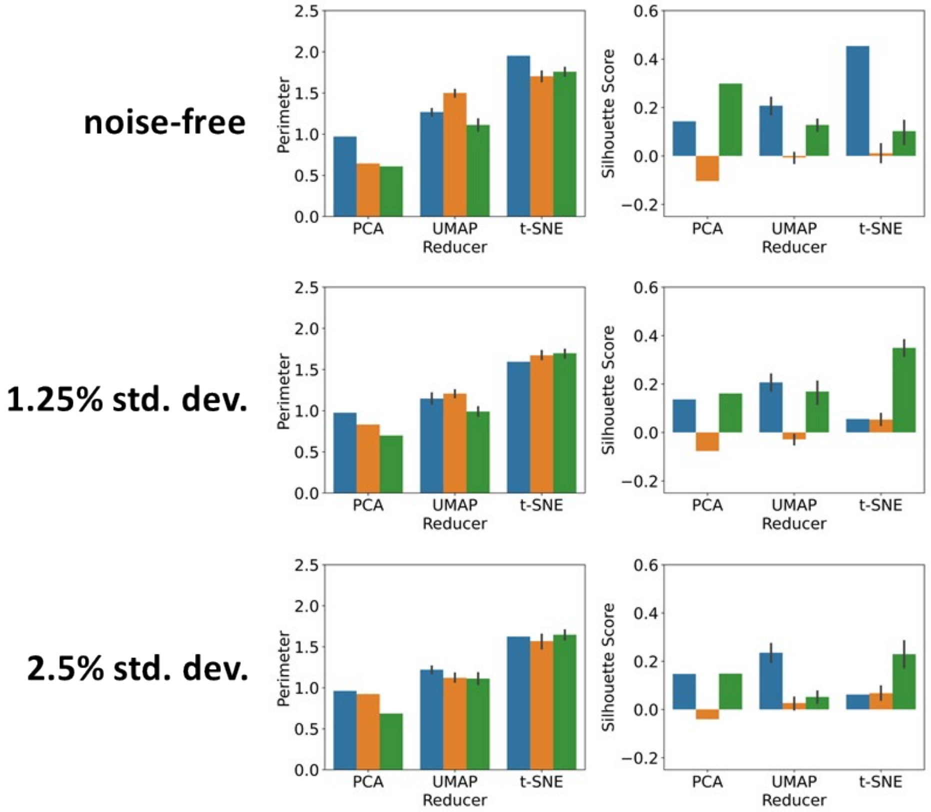


FIG S12 Comparison of the overall classification results for PCA, UMAP and t-SNE in the presence of small to moderate amounts of electric noise, based on average P- and S scores, applied to: numerical data (blue), image data (orange); and feature extractor output (green).

For UMAP, we observe a similar picture, even at the highest noise level, its performance when applied to features drops significantly. However, the most drastic effect is observed for t-SNE. Initially performing strongest when applied to numerical data, the *S*-score drops dramatically even for a relatively small amount of noise. Identifying the reasons for this effect requires further study. On the other hand, its application to features leads to a rather robust performance and for the highest noise level, t-SNE on features is best performing, jointly with UMAP on data.

References

[1] Wei, Y., Sun, Y. & Tang, X. Autoacceleration and kinetics of electrochemical polymerization of aniline. J. Phys. Chem. 1989, 93, 4878–4881.

[2] Mohilner, D. M., Adams, R. N. & Argersinger, W. J. Investigation of the kinetics and mechanism of the anodic oxidation of aniline in aqueous sulfuric acid solution at a platinum electrode. J. Am. Chem. Soc. 1962, 84, 3618–3622.