Supplementary Material: An extended auto-encoder model for reaction coordinate discovery in rare event molecular dynamics datasets

M. Frassek,[1] A. Arjun,[1] and P.G. Bolhuis[1, a)]
van 't Hoff Institute for Molecular Sciences, University of Amsterdam, PO Box 94157, 1090 GD Amsterdam, Netherlands

Appendix A: Data visualization

1. Visualising model outputs

After the model was tuned and trained, it was possible to gain new insights from it. The raw outputs of the model, however, lacked interpretability. Therefore, methods were written to allow a visual inspection of the model outputs.

Plotting committor predictions To visualise the committor prediction, a set of pairwise heat maps was generated, covering all pairs of dimensions that can be selected from the dataset. Representative snapshots were generated for each of the heat maps. For each point on the heat map, the representative snapshot consisted of fixed values for the heat map's x and y dimension, given by the position of the point on the grid, as well as representative values for the other dimensions. The representative values of these other dimensions were calculated as the mean values of all points with the given values for the dimensions of interest, as depicted in formula A1

$$\bar{x}_i = \frac{\sum_{j=1}^{n} x_{i_j}}{n} \tag{A1}$$

where $\bar{x}_i$ is the calculated value for dimension i and n is the number of points that match the current values of the dimensions of interest.
To generate the heat maps each grid point's representative snapshot was passed through the committor predictor and the predicted committor taken as the value displayed in the heat map.

Plotting input reconstruction To visualise the reconstruction, a set of scatter plots was generated. Each scatter plot displayed the input along one dimension on the x-axis and the corresponding prediction of the base autoencoder along the y-axis.
Similar to the visualisation of the committor predictions, the value for the dimension of interest was defined by the position on the x-axis of the scatter plot, while the values for the other dimensions were calculated via formula A1.

Mapping paths onto the latent space By giving a snapshot to the encoder, it was possible to map it onto the latent space. Similarly, complete paths could be mapped onto the latent space, by encoding all its snapshots. To gain insights into the representation of the configurational space on the latent space, a method was written that selected paths from the dataset, mapped them on the latent space, and plotted these latent paths.

Mapping paths onto the committor space As for the encoder, the committor predictor not only allowed mapping single snapshots to the committor but also mapping of whole paths onto the committor space. A method was written to select paths and plot the development of the committor over their course.

Mapping latent paths onto the configurational space Not only mappings from the configurational space were possible, but the reconstruction decoder also allowed a mapping from latent space to configurational space. By defining any path on the latent space a corresponding path on the configurational space would be reconstructed. A method was written to generate a representative latent path by interpolating between the maximum and minimum values of the latent space variable and plot the resulting reconstructed path.

2. Visualising the ground truth

To allow for a visual comparison between prediction and ground truth, 2D heat maps for each pair of dimensions were generated from the original dataset. Similar to the p_B approximation, a weighted mean associated with each grid point was calculated from the labels and weights. Here, however, the snapshots were binned along only two dimensions, at each step.

Appendix B: Datasets and data processing for hydrate nucleation

1. Datasets

TIS data The TIS dataset was taken from[1]. The TIS was performed at T = 280 K and P = 500 bar, on a cubic box containing 2,944 water and 512 methane molecules. For the water molecules, the Tip4P/ICE force field was used, while the methanes were modelled via united atom Lennard-Jones interactions. The number of mutually coordinated guests (MCG), as an indicator of the nucleus size, was chosen as the OP for interface definition. Interfaces were placed at MCG = 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90 and 100. Paths belonging to the ten highest interfaces were used in this work, while the others were disregarded.

TPS data The TPS data was produced as described in[2], at T = 280 K with spring shooting moves, a spring

a)Electronic mail: p.g.bolhuis@uva.nl

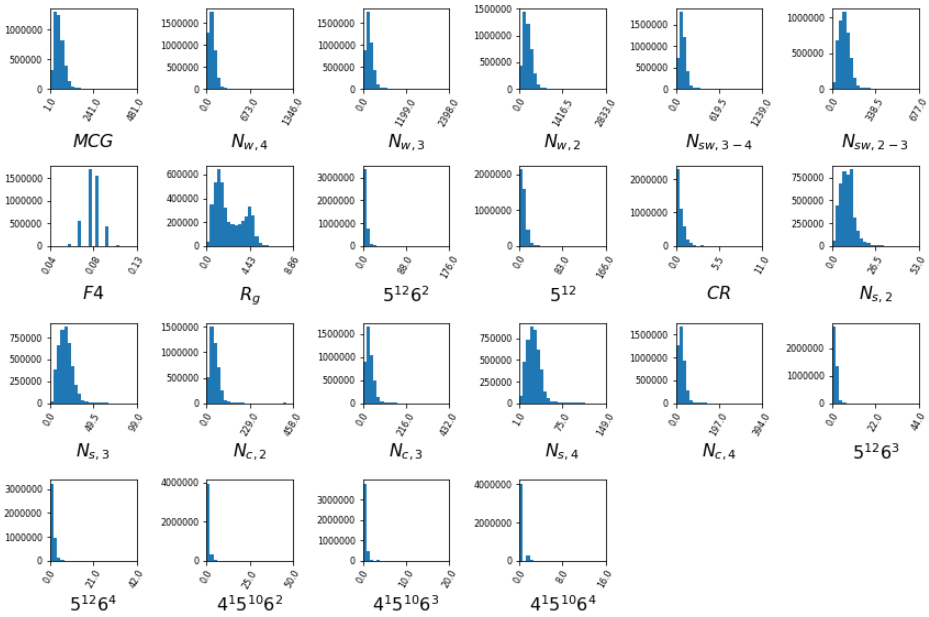

FIG. 1. Nucleation data. Distribution of values for all order parameters.

constant of 0.1, and a frame shift of 200. 153 AB paths were generated this way and added to the dataset used in this work.

2. Data import and preprocessing

As a first step, the raw data was imported. For the toy datasets, paths and the corresponding labels were read from files. Since the dataset already represented the RPE, all paths were assigned the same weight.
For the methane hydrate nucleation data, paths were also read from the raw TIS and TPS files. The label of each path was chosen based on the states of its first and last snapshot. Paths starting or ending outside of these state definitions were removed from the dataset. AA paths were associated with the label 0 and AB paths with the label 1.
Since the distribution of different paths among the TIS and TPS data was biased, different weights were used to unbias the data. By applying a reweighting process[3] to the TIS data, each interface was assigned an RPE weight. These RPE weights corrected for the over- or under-representation of paths belonging to the given interface when compared to an unbiased path ensemble. Further, each TIS path was associated with a Monte Carlo weight which corrected for the undersampling of that specific path (that is, the amount rejection of the trial paths before a new path is accepted), and which was provided with the path data. A path's final weight was calculated by reading its respective Monte Carlo weight from the raw TIS files and multiplying it with the RPE weight associated with the path's interface.

We would like to augment the TIS dataset with the reactive TPS trajectories. However, paths from the TPS raw files were extracted without associated RPE weights, and could therefore not be directly added to the TIS set. To do so, we need to assign the correct path probabilities to the TPS paths. Since in the high interfaces, i.e MCG100, a large percentage of paths lead to the final state, these are effectively reactive transition paths. We therefore took the total probability of the reactive paths in the MCG100 interface, and redistributed this probability over the TIS and TPS sets of reactive paths. That is, both the reactive TPS and TIS paths in the nMCG100 interface receive identical weights w_{new} given by

$$w_{new} = w_{MCG_{100}} \frac{n_{MCG_{100}}}{n_{MCG_{100}} + n_{TPS}} \tag{B1}$$

where $w_{MCG_{100}}$ is the RPE weight for reactive paths associated with the highest TIS interface, $n_{MCG_{100}}$ the number of paths in the highest TIS interface, and n_{TPS} the number of TPS paths. This automatically ensures that the augmented set of $nMCG100 + nTPS$ path have the same total probability in the date set as the original TIS reactive paths.

3. Nucleation data characterisation

Figure 1 depicts the distribution of values for the different order parameters in the full methane hydrate nucleation dataset.

Figure 2 shows the distribution of approximated p_B values for the full dataset. Points with $p_B = 0$ took up by far the biggest fraction of all points. Points where

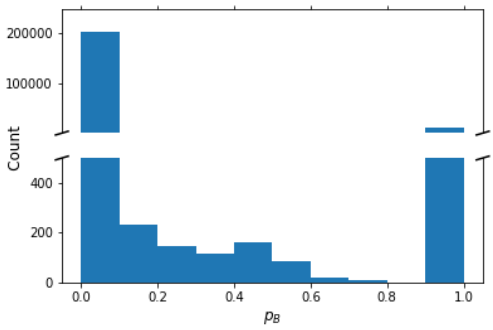

FIG. 2. Nucleation data. Distribution of approximated p_B values for the full-dimensional dataset

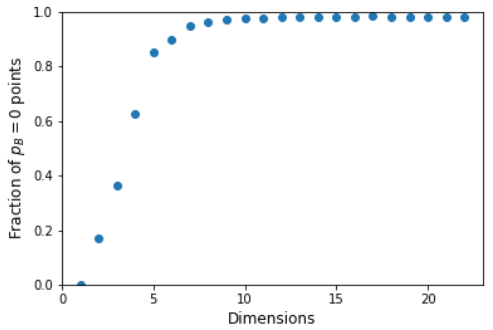

FIG. 3. Fraction of points where $p_B = 0$ among all points calculated for different numbers of datasets with different numbers of dimensions.

$p_B = 1$ occurred less often but still overshadowed the number of points with $0 < p_B < 1$. For this range only comparatively few points existed. Figure 3 plots the fraction of points with $p_B = 0$ depending on the number of dimensions of the dataset. When only few dimensions were used, only a small fraction of points displayed a p_B of 0. For higher dimensionalities, however, this fraction increased. For eight or more dimensions, almost all points were be associated with a p_B of 0.

Appendix C: Parameter settings tested

To gain the best predictions from the model, different parameter settings were tested.

1. Activation functions

The activation functions within the encoder, reconstruction decoder and prediction decoder could all be selected to be either linear or non-linear, allowing for eight different combinations. Figure 4 displays the predictions made for the Z-potential with the eight different variants, while figure 5 shows the input reconstruction of these variants.

The only predictions that resemble the underlying Z-potential closely were produced by the $E_N C_N R_L$ and $E_N C_N R_N$ variants. The $E_L C_N R_L$ and $E_L C_N R_N$ variants also captured the trends approximately but were not fully able to reproduce the underlying mechanism as they can only produce a linear dividing surface. The other variants produced poor predictions.
Of the $E_N C_N R_L$ and $E_N C_N R_N$ variants, neither produced a perfect reconstruction of the x_1 and x_2 inputs, but the $E_N C_N R_N$ model produced the better reconstruction of the two. It was therefore chosen as the default setting for the model. The $E_L C_N R_L$ variant did produce an almost perfect reconstruction of the input of these two dimensions, but due to its failure to learn non-linear dividing surfaces, overall it performed worse than the $E_N C_N R_N$ variant.

2. Outlier cutoffs

Outliers, snapshots that are at the outside of the relevant variable range, were removed in order to prevent having too many empty bins in areas where the data is just too unreliable. When removing outliers, values below and above given thresholds were capped. Figure 6 displays the ground truth of the methane hydrate nucleation dataset with different such thresholds.

For the 10th/90th percentile, the barrier was not captured, while the 0.5th/99.5th percentile and 0.1st/99.9th percentile captured a region extending far beyond the barrier. For the 2nd/98th percentile, the barrier was nicely captured while not including too much of the post-critical region that contains little additional information. Therefore these percentiles were chosen as the caps for the outlier removal.

As a general rule of thumb for choosing these caps, the outlier removal should not remove the transition from the data set. On the other hand, the outlier removal should lead to sufficient data points in most bins of the variable range.

3. Evaluated labels and weights

As data imbalance can interfere with the learning of neural networks, different schemes to balance the dataset were tested:
standard RPE weights: The weights originally associated with each data point. Calculated by reweighting the dataset.
equal weights: All data points were assigned the same weights, independent of any of their properties.
p_B-balanced weights: Data points were binned according to their p_B and each bin assigned the same total weight. All points within a bin then got assigned an equal share of the bin's total weight. Data points with a p_B

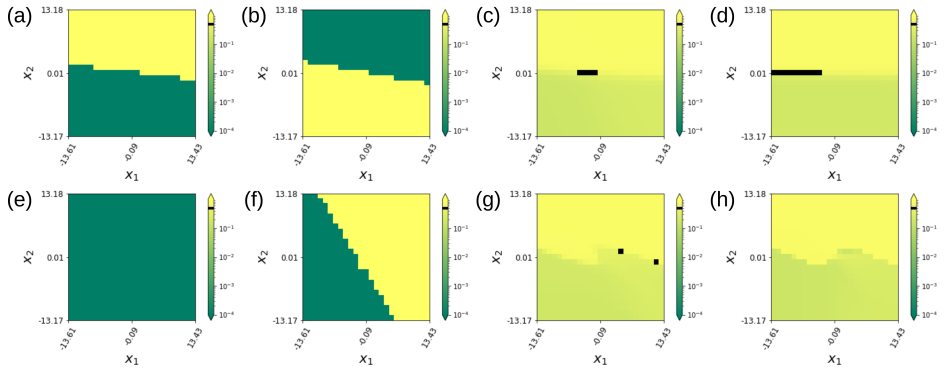

FIG. 4. Predictions made by the model for the Z-potential dataset based on activation functions (linear (L), non-linear (N)) used for its different submodels (encoder (E), committor decoder (C), reconstruction decoder (R)): (a) $E_LC_LR_L$ (b) $E_LC_LR_N$ (c) $E_LC_NR_L$ (d) $E_LC_NR_N$ (e) $E_NC_LR_L$ (f) $E_NC_LR_N$ (g) $E_NC_NR_L$ (h) $E_NC_NR_N$

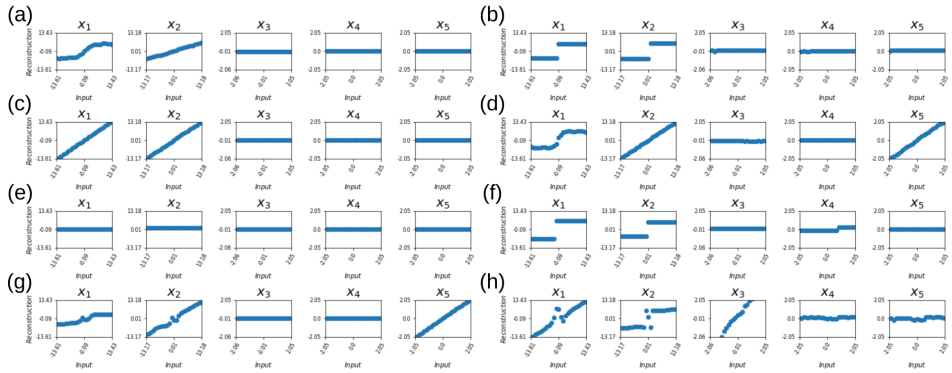

FIG. 5. Input reconstructions produced by the model for the Z-potential dataset based on activation functions (linear (L), non-linear (N)) used for its different submodels (encoder (E), committor decoder (C), reconstruction decoder (R)): (a) $E_LC_LR_L$ (b) $E_LC_LR_N$ (c) $E_LC_NR_L$ (d) $E_LC_NR_N$ (e) $E_NC_LR_L$ (f) $E_NC_LR_N$ (g) $E_NC_NR_L$ (h) $E_NC_NR_N$

occurring more frequently were therefore deemphasised, while data points with rare p_B values were emphasised.

hypercube-balanced weights: Data points were binned according to their position in the configurational space and each bin assigned the same total weight. All points within a bin then got assigned an equal share of the bin's total weight. Data points of highly occupied regions were therefore deemphasised, while data points of regions with fewer points were emphasised.

multi-dimensionally-balanced weights: For each dimension, data points were binned according to their position on that dimension, the bins assigned equal weights, and the weights per bin shared among its members. A point's final weight was calculated by multiplying the dimensionally-balanced weights for all its dimensions. Data points at highly occupied regions of one or more dimensions were therefore deemphasised, while points at less occupied regions of all their dimensions were emphasised.

Fig. 7 shows the predictions made by the model for the nucleation dataset based on the composition of the tensorflow dataset used for training. In this dataset components were varied: Labels (L), committor prediction weights (CW), reconstruction weights (RW). Within these categories, there were various settings: Labels: standardlabels (s), approximatedpBs (pB) - Weights: standard RPE (s), equal (e), pB-balanced (pBb), hypercube-balanced (hcb),multi-dimensionally-balanced (mdb). In Fig. 7 eight different combination were shown.

$L_sCW_sRW_s$ caused the model to overfocus on the more abundant data points with a label of 0, fully overshadowing any points with a label of 1. This arrangement was therefore unsuited for the ML approach.

$L_{p_B}CW_{mdb}RW_{mdb}$, $L_{p_B}CW_{p_Bb}RW_{hcb}$, $L_{p_B}CW_{p_Bb}RW_{mdb}$, and $L_{p_B}CW_{p_Bb}RW_{p_Bb}$, produced better predictions which also captured the barrier. However, the range of predicted committor values was narrow, not capturing the low or high p_Bs well.

$L_{p_B}CW_{hcb}RW_{hcb}$, $L_sCW_eRW_e$, and $L_{pB}CW_eRW_e$ all produced good predictions where the barrier as well as the lower and higher p_B regions were captured well. While all three of these compositions might have been suitable, $L_{pB}CW_eRW_e$ yielded the best reproduction of

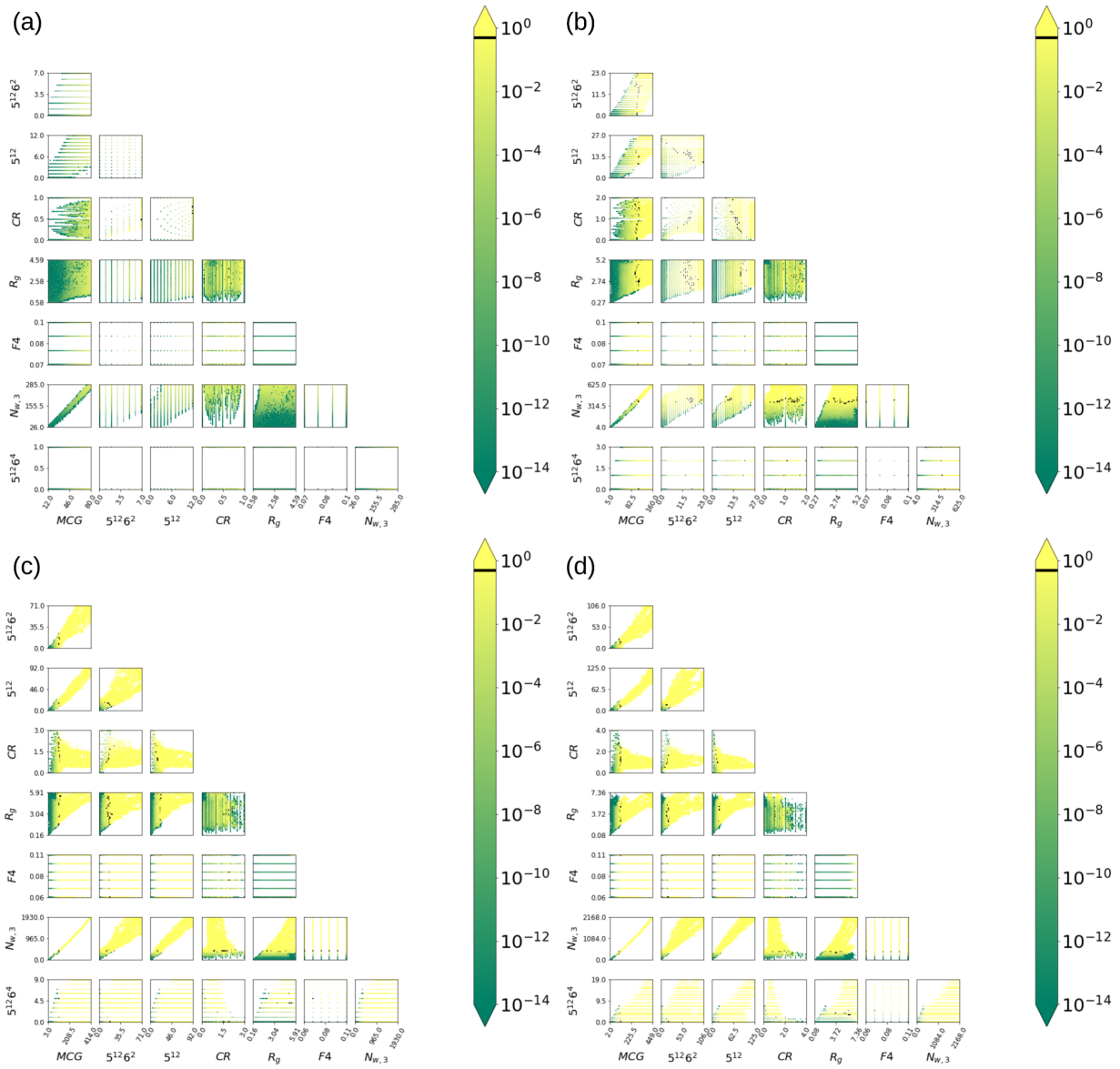

FIG. 6. Pairwise heat maps for eight representative dimensions of the nucleation dataset. Ground truths after processing with different outlier cutoff thresholds: (a) 10th and 90th percentile (b) 2nd and 98th percentile (c) 0.5th and 99.5th percentile (d) 0.1st and 99.9th percentile

the lower p_B regions. Therefore the datasets for training the model were composed of the approximated p_Bs and all points were given equal weight.

4. Loss functions

Different loss functions for the committor predictor were tested for their suitability (see Figure 8). These entailed the absolute error, squared error, binomial log likelihood, binary log likelihood, difference of logs, and logarithmic absolute error (Eq. C1 - C6).
Absolute error:

$$L = \text{abs}(z_{actual} - z_{pred}) \tag{C1}$$

Squared error:

$$L = (z_{actual} - z_{pred})^2 \tag{C2}$$

Binary log likelihood:

$$L = -(z_{actual}\log(z_{pred}) + (1 - z_{actual})\log(1 - z_{pred})) \tag{C3}$$

Binomial log likelihood:

$$L = -(2z_{actual}\log(z_{pred}) + 2(1 - z_{actual})\log(1 - z_{pred})) \tag{C4}$$

Difference of logs:

$$L = \text{abs}(\log(z_{actual}) - \log(z_{pred})) \tag{C5}$$

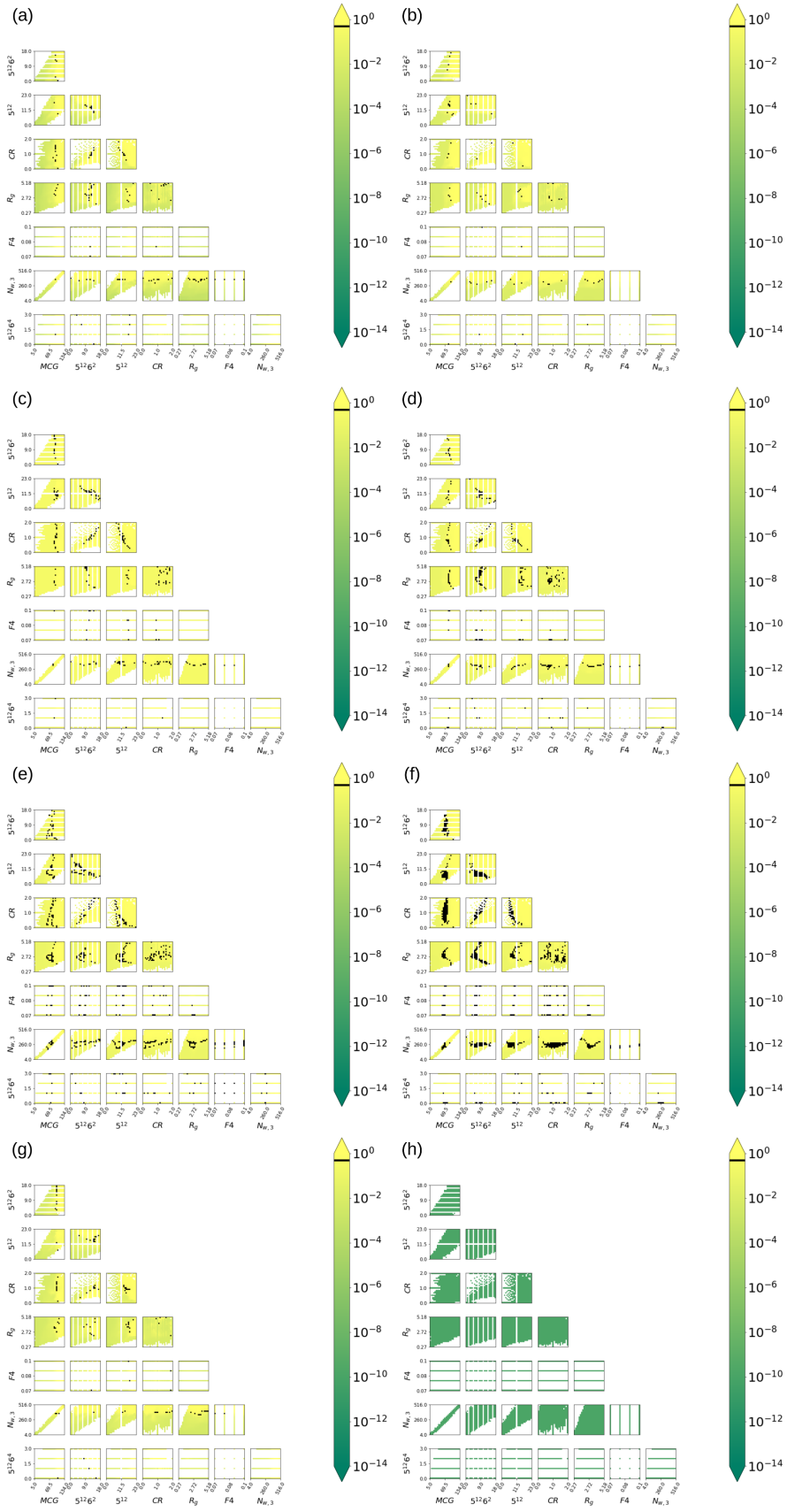

FIG. 7. Predictions made by the model for the nucleation dataset based on the composition of the tensorflow dataset used for training - Dataset components: Labels (L), committor prediction weights (CW), reconstruction weights (RW) - Labels: standard labels (s), approximated p_Bs (p_B) - Weights: standard RPE (s), equal (e), p_B-balanced ($p_B b$), hypercube-balanced (hcb), multi-dimensionally-balanced (mdb): (a) $L_{p_B} CW_e RW_e$ (b) $L_{p_B} CW_{hcb} RW_{hcb}$ (c) $L_{p_B} CW_{mdb} RW_{mdb}$ (d) $L_{p_B} CW_{p_B b} RW_{hcb}$ (e) $L_{p_B} CW_{p_B b} RW_{mdb}$ (f) $L_{p_B} CW_{p_B b} RW_{p_B b}$ (g) $L_s CW_e RW_e$ (h) $L_s CW_s RW_s$

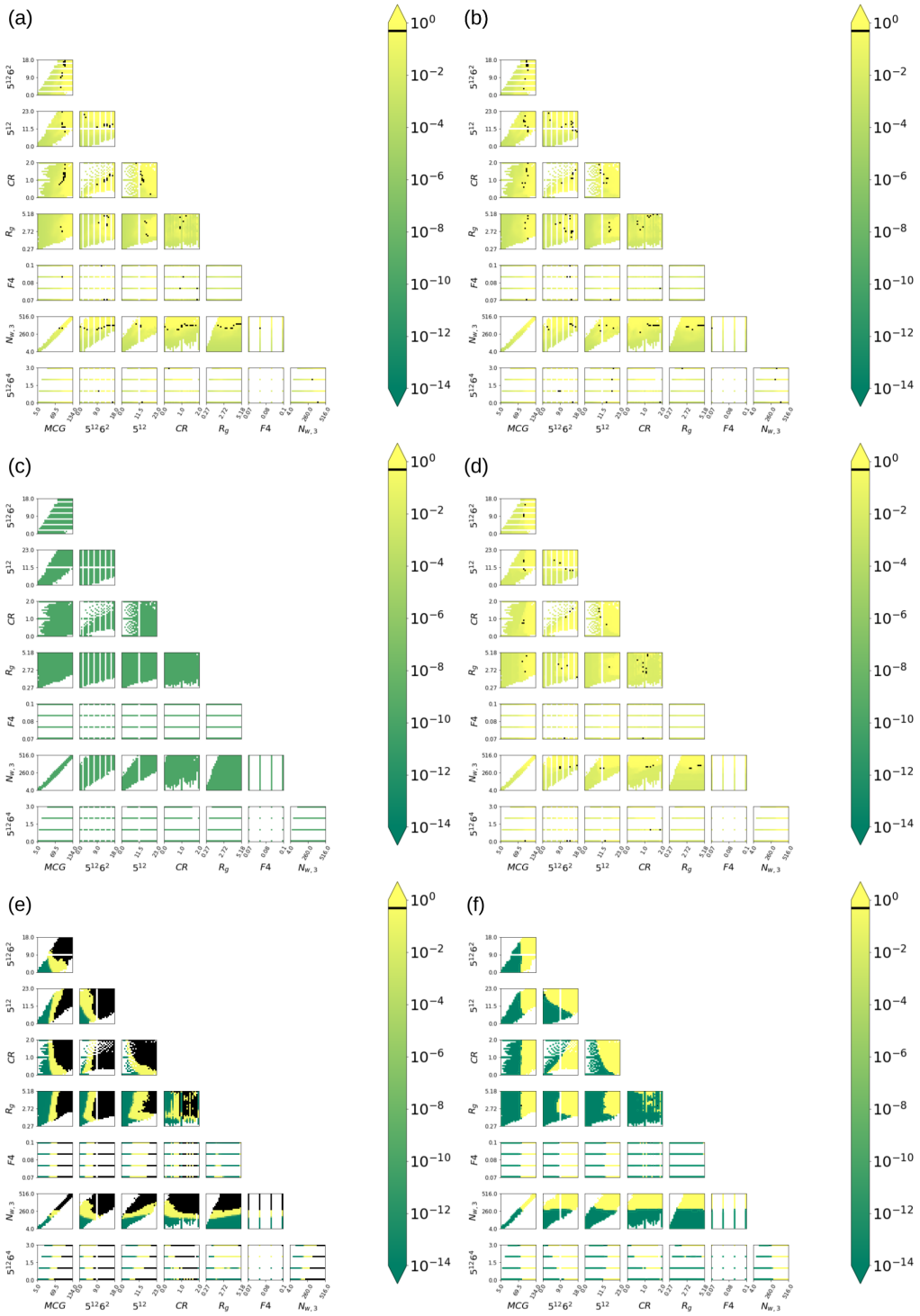

FIG. 8. Committor predictions made by the model for the nucleation dataset based on the loss function chosen for training (a) binary log likelihood (b) binomial log likelihood (c) absolute error (d) squared error (e) difference of logs (f) logarithmic absolute error

Logarithmic absolute error:

$$L = \log(\text{abs}(z_{actual} - z_{pred})) \tag{C6}$$

When testing the EAE with these different loss functions for the committor decoder, a wide variety of behaviour was found.

Both the binary log likelihood and the binomial log likelihood function (Eq. C3 and C4) yielded good results for the predictions. The barrier was captured and both grid

points with low and high p_Bs were reproduced. While the regions with very low p_B values were not replicated perfectly, the overall performance of the model with these loss functions was deemed good.

Using the absolute error (formula C1) caused the model to predict low p_B values for all input values and to fail at capturing a barrier. This loss function was therefore deemed ill-suited for our model. When using the squared error loss function (formula C2), a similar behaviour of the model was observed as for the binomial or binary log likelihood function. However, the predicted barrier region appeared to be sparser with more quick transitions from p_B values below 0.5 to p_B values above 0.5. While the performance of the model with this loss function was decent, the binary or binomial log likelihood were considered to be more suitable.

The difference of logs (formula C5) was originally selected to emphasise the lower p_B regions. This emphasis, however, appears to have caused the model trained with this loss function to overfocus on these lower regions. While the model succeeded in capturing the regions with very low committor values, the remainder of the predictions were poor. Barriers were predicted by the model, but instead of being a narrow line, they spanned large parameter ranges. The model's predictions appear to have flattened out at values around $p_B = 0.5$ and never reached higher predictions. Despite capturing the low p_B regions better than the other loss functions, its failure to capture regions of higher p_B correctly caused the difference of logs functions to be unsuited for our model.

As the difference of logs, the logarithmic absolute error function (formula C6) was chosen to emphasise the regions with lower p_B values. However, the model trained with this loss function experienced similar problems to the one trained with the difference of logs function. While regions with lower committor values were nicely captured, the model failed to reproduce regions with higher committor values. This model even failed to predict barriers with its highest predicted p_B values lying below 0.5. Due to its inability to capture the barrier and higher p_B values, the logarithmic absolute error was also ill-suited for this work.

Overall, the binary log likelihood and the binomial log likelihood function yielded the best predictions. As they were both comparable, the binary negative likelihood was chosen arbitrarily between the two to serve as the model's default loss function.

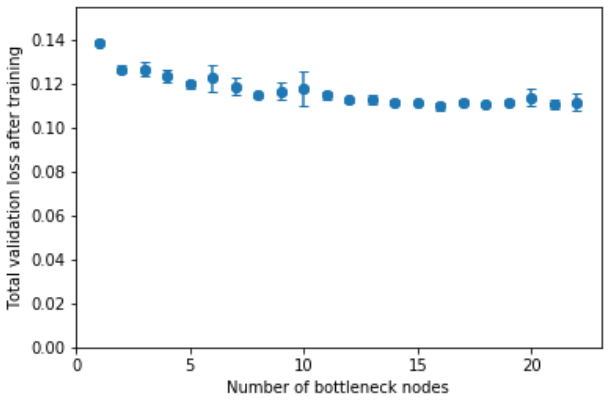

FIG. 9. Mean loss after training the model on the nucleation dataset based on the number of nodes in the bottleneck layer. Mean losses were calculated from three separately trained models.

5. Bottleneck size

The size of the bottleneck impacts how much information can pass through it. A higher bottleneck size should allow for more information to be passed to the decoder and thereby reduce the loss. A larger bottleneck size, however, also means that information is less condensed and the latent space encoded in the bottleneck therefore less informative. Figure 9 plots the model loss against the size of the bottleneck.

While an increasing number of bottleneck nodes yielded a lower loss, the gains per additional bottleneck node were minuscule. Therefore a bottleneck size of 1 was chosen for this work, as this was expected to yield the most condensed latent space at the cost of only a small drop in performance.

Appendix D: Additional results

1. Full dimensional dataset

Figure 10 and 11 show the predictions of the EAE model for the full-dimensional methane hydrate nucleation dataset.

As for the predictions made for only eight selected dimensions, the models predicted a clear barrier in many of the heat maps, while some dimensions were too sparse to make out clear trends. The predictions made by the nonlinear and the linear encoder variant appear to agree in the position of the barriers along the different dimensions, although the predicted barriers differ to some extent in shape.

2. Shooting data

The set of shooting points used in Ref.[2] was plotted in Fig. 12.

For a direct comparison, figure 13 shows the $N_{w,3}/5^{12}6^2$ plot for both the full dataset and shooting points, including the respective dividing surfaces predicted in Ref.[2].

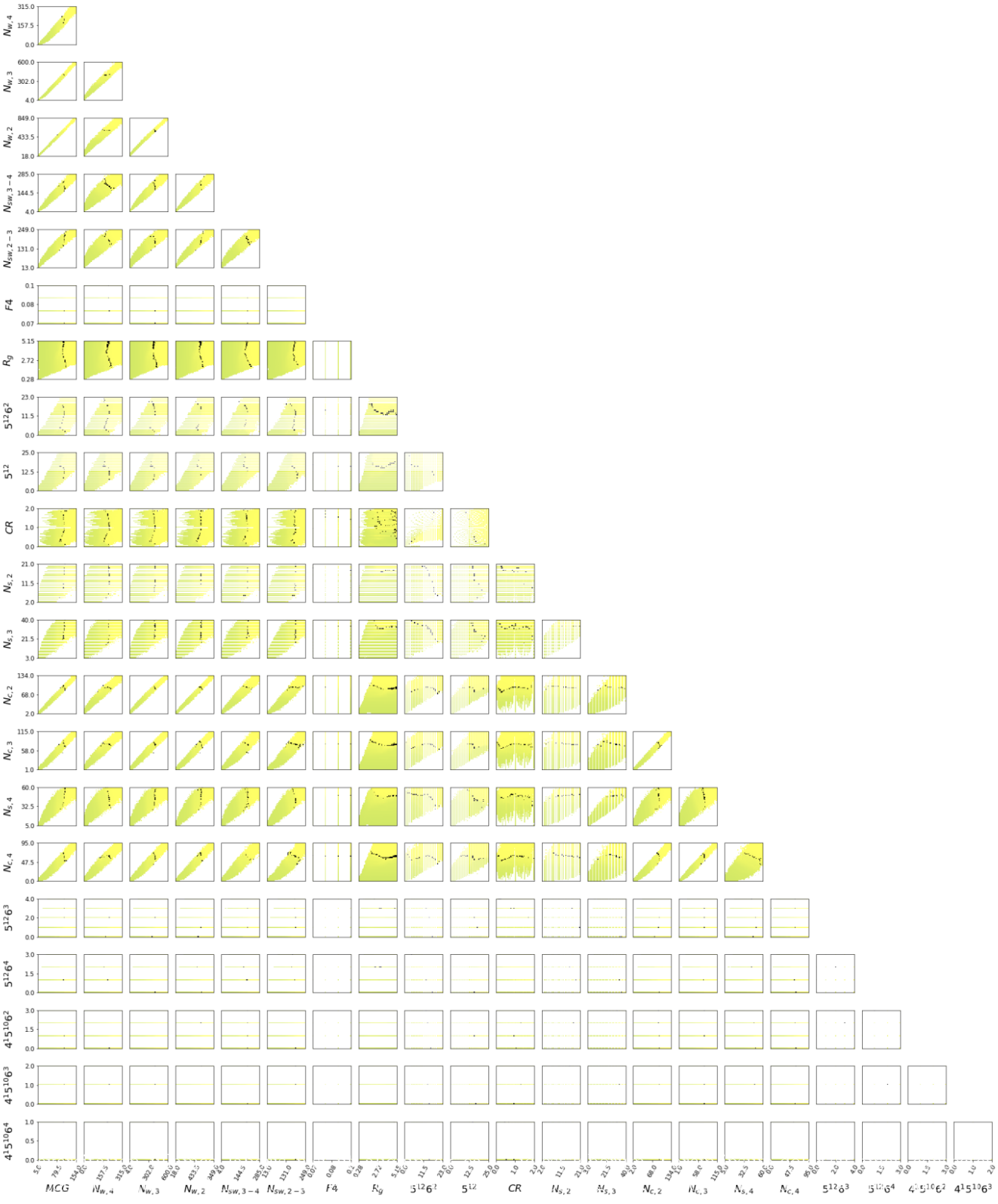

FIG. 10. Nucleation data, non-linear encoder: Pairwise heat maps for all 22 dimensions. Prediction, made by the model after training.

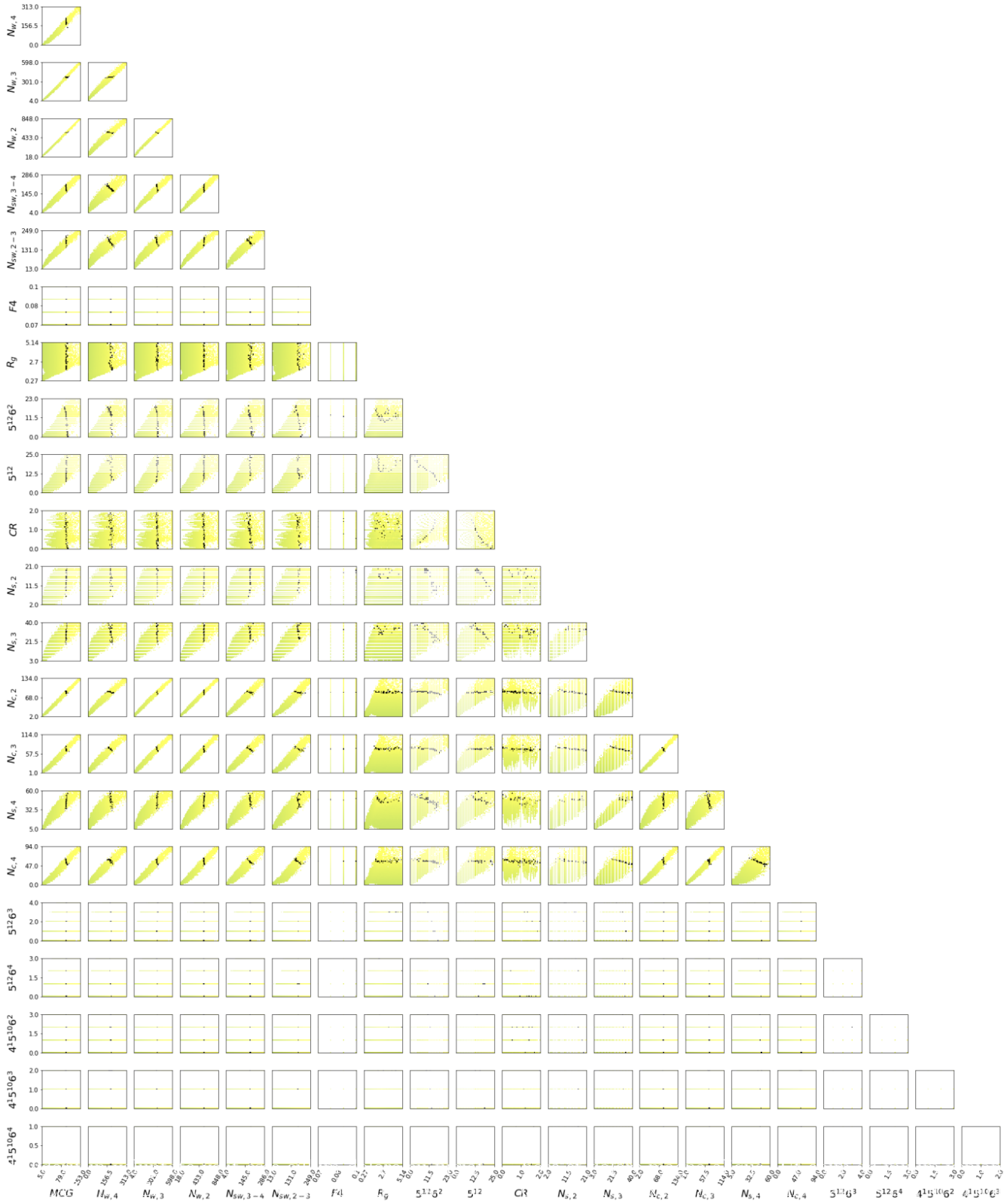

FIG. 11. Nucleation data, linear encoder: Pairwise heat maps for all 22 dimensions. Prediction, made by the model after training.

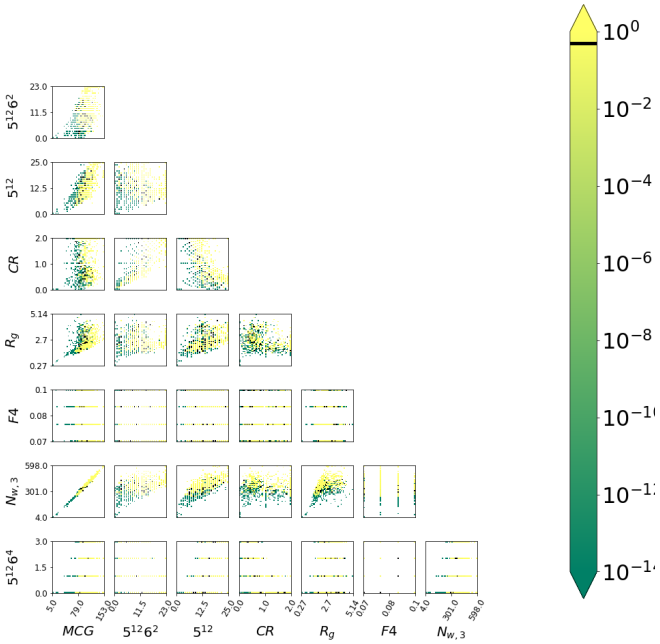

FIG. 12. Nucleation data: Pairwise heat maps for eight representative dimensions. TPS shooting point data. (Data taken from Ref.[2])

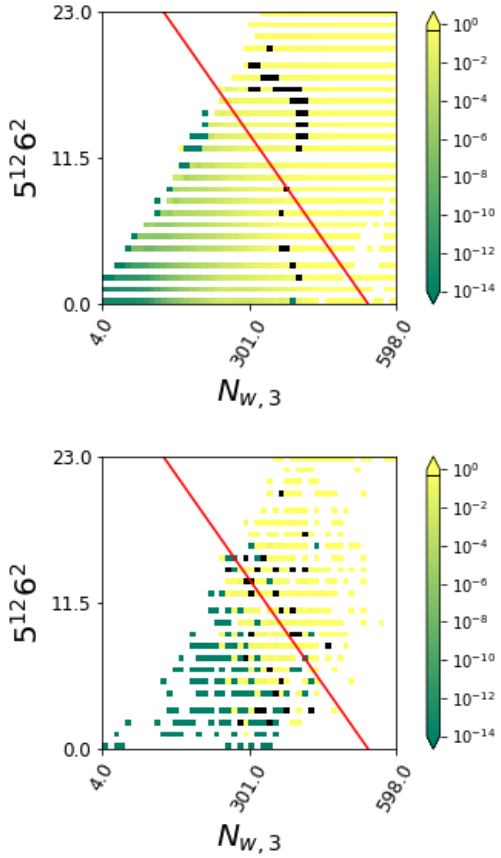

FIG. 13. Nucleation data, ground truth $N_{w,3}/5^{12}6^2$ plot. The red line shows the dividing surface determined in Ref.[2] (Top) TPS and TIS dataset (Bottom) Shooting point dataset.

For the shootings points, the dividing surface defined by the black points and the previously determined dividing surface, represented by the line, match closely. For the TPS/TIS data the dividing surfaces match less closely. These two datasets, therefore, seem to indicate different positions of the dividing surface. These differences, however, are not substantial.

[1]A. Arjun and P. G. Bolhuis, J. Phys. Chem.B **124**, 8099 (2020).
[2]A. Arjun, T. Berendsen, P. G. Bolhuis, *et al.*, Proc. Nat. Acad. Sci. USA **116**, 19305 (2019).
[3]J. Rogal, W. Lechner, J. Juraszek, B. Ensing, and P. G. Bolhuis, J. Chem. Phys. **133**, 174109 (2010).