Machine learning models for Si nanoparticle growth in nonthermal plasma

Matt Raymond[1], Paolo Elvati[2], Jacob C. Saldinger[*3], Jonathan Lin[1], Xuetao Shi[†2] and Angela Violi[‡1,2,3]

[1]Mechanical Engineering, University of Michigan, Ann Arbor, 48109-2125, Michigan, USA
[2]Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, 48109-2125, Michigan, USA
[3]Chemical Engineering, University of Michigan, Ann Arbor, 48109-2125, Michigan, USA

October 31, 2024

Abstract

Nanoparticles (NPs) formed in nonthermal plasmas (NTPs) can have unique properties and applications. However, modeling their growth in these environments presents significant challenges due to the non-equilibrium nature of NTPs, making them computationally expensive to describe. In this work, we address the challenges associated with accelerating the estimation of parameters needed for these models. Specifically, we explore how different machine learning models can be tailored to improve prediction outcomes. We apply these methods to reactive classical molecular dynamics data, which capture the processes associated with colliding silane fragments in NTPs. These reactions exemplify processes where qualitative trends are clear, but their quantification is challenging, hard to generalize, and requires time-consuming simulations. Our results demonstrate that good prediction performance can be achieved when appropriate loss functions are implemented and correct invariances are imposed. While the diversity of molecules used in the training set is critical for accurate prediction, our findings indicate that only a fraction (15-25%) of the energy and temperature sampling is required to achieve high levels of accuracy. This suggests a substantial reduction in computational effort is possible for similar systems.

Keywords: Molecular Dynamics, sticking coefficient, silane, machine learning, nanoparticle, nonthermal plasma

1 Introduction

Nonthermal plasmas (NTPs) are unique environments where low-temperature neutral species and ions coexist with high-temperature electrons. For this reason, these systems have received considerable attention, especially for synthesizing particles and nanoparticles with significant tunability. This flexibility in the final particle properties results from an environment with enough localized energy to cross relatively high free energy barriers while avoiding excessive thermal energy and discouraging agglomeration [1]. As a result, the synthesis of nanoparticles and thin films under these conditions holds potential applications in biomedicine [2, 3], energy [4–6], microelectronics [7], and catalysis [8].

However, modeling these environments remains a significant challenge due to the combined non-equilibrium and multiscale nature. [9–11] Even when narrowing the description only to a specific scale or class of processes, such as particle growth (*e.g.*, nucleation, coagulation, surface deposition) [12–14], the accuracy of the methods depends on their ability to model a variety of size- and charge-dependent growth mechanisms. These processes, in turn, depend on the propensity of species, generally radicals, to form stable bonds upon collision with other particles or surfaces. Still, these processes have been frequently estimated using fixed values, independent of the colliding species and energies [14], primarily due to the complexity of obtaining a more detailed functional form. Recently [15], we have shown how atomistic simulations can capture the complex reactivity of small neutrals and provide parameters that can be used in reactor models [16, 17]. While these previous and current works focus on silane particles, the underlying methodology is general and adaptable to various conditions where species' internal and translational energy distributions differ. This flexibility sets our method apart from others, such as the one recently published by Bal and Neyts [18],

*Now at Low Carbon Pathway Innovation at BP
†Now at the Dana-Farber Cancer Institute at Harvard
‡E-mail: avioli@umich.edu

which (among other differences) does not make assumptions about the translational energy distribution or the specific reaction under investigation. Due to the large size of some involved species and the resulting (lack of) separation of vibrational modes, we observed a variety of competing reaction mechanisms involving a complex interplay of physisorption, chemisorption, and desorption.

While very informative, deriving these reacting probabilities via molecular dynamics (MD) simulations remains time-consuming and computationally burdensome. Even when using classical reactive MD, a timestep of the order 10 as is required to guarantee correct numerical integration of the equations of motion during the reactions. Moreover, due to numerous variables (*e.g.*, impact parameter, speed distribution, surface composition) and relevant species present in such reactive systems, the number of simulations required to capture the collisions experiences rapid combinatorial growth. More effective means of deriving the collision parameters must be considered to scale this approach. In this work, we focus on machine learning (ML) methods, which offer the potential to formulate a dependency between the system conditions and the final collision outcome. Data-driven methods do not remove the need for MD simulations but allow for a drastic reduction in computational effort.

Recent works have used ML methods to overcome similar combinatorial problems associated with the growth of nanoparticles in reactive gas-phase environments, accurately predicting the aggregation propensities of soot precursors [19]. However, no existing work addresses the same scientific questions in the context of nonthermal plasmas. Most studies focus on predicting plasma properties [20–22] and plasma-surface interactions, from surface deposition [11, 23–26], plasma etching [27–29], and surface modification [30]. In contrast, this work examines a scale between detailed individual reactions and simplified larger systems, where detailed chemistry must be approximated. Our focus is on particles approximately 1 nm in diameter colliding with small reactive fragments (SiH_y and Si_2H_y). Building upon previous data [15], we demonstrate how easily computable properties can be used to train ML models to generate predictions for new species or mitigate the MD computational cost.

2 Methodology

2.1 Molecular Dynamics Simulations

We performed classical reactive molecular dynamics simulations to study the collisions between disilanes, Si_2Hx, and other silane clusters and molecules using the same procedure described previously [15]. First, we independently equilibrated both colliding species' rotational and vibrational modes, and then we performed microcanonical simulations at a fixed impact velocity. Between 40 and 100 different collision vectors were imposed to parallel the line passing through the center of mass of the two species (*i.e.*, impact parameter = 0 or, equivalently, impact angle = π). Simulations were performed using LAMMPS [31] using the ReaxFF force field [32] in combination with a dynamic charge equilibration model [33] and integrated the equations of motion every 0.01 fs. To analyze the collision outcome, we monitored the minimum distance between all the atoms or only the Si atoms of each cluster.

The conformations of the colliding species were generated from the canonical simulations at 300 K, 400 K, 500 K, 600 K, and 900 K. Properties are computed by reweighting the collisions using a Maxwell-Boltzmann distribution, which allows labeling each system with a single temperature. For clarity, we grouped the colliding species in two sets, labeled "clusters" and "impactors", but there is no physically meaningful distinction associated with each set. Molecules in the cluster set are silanes that cover different sizes and H-coverage (*i.e.*, Si_2H_6, Si_4, and $Si_{29}H_x$ with $x = 18, 27, 31, 36$), while as impactors, we considered different disilanes (*i.e.*, Si_2H_x with $x \in [1, 6]$) in all possible hydrogen distribution (*e.g.*, for Si_2H_4, we simulated both $H_2Si^{\bullet}-{}^{\bullet}SiH_2$ and $HSi^{\bullet\bullet}-SiH_3$). The results of these simulations were combined with previously computed data [15] to create a dataset of 390 collision pairs based on approximately 650 000 simulations.

2.2 Machine Learning

We compared the predictive performances of seven standard ML models for predicting sticking probabilities: an unpenalized linear model, ElasticNet, Kernel Ridge Regression (KRR), Support Vector Regression (SVR), k-nearest neighbors (KNN) [34], DeepSets [35], and Light Gradient-Boosting Machine (LGBM) [36].

2.2.1 Input features

For model input, we generated feature vectors using parameters that describe properties likely to affect the sticking probability for silane molecules [15] (*i.e.*, H coverage, temperature, and molecule's size). Specifically, for each cluster and impactor, we used the number of Si atoms, H atoms, and a vector of the number of unpaired electrons per Si atom (to differentiate between isomers). For the disilanes, this two-dimensional vector indicates the number of unpaired electrons on each Si atom; in contrast, only the total number was used for the larger clusters, and the second element was always set to 0. For particle a, we denote this feature vector as $\boldsymbol{f}_a \in \mathbb{R}^4$. Each particle interaction has an associated translational temperature, denoted $t \in (0, \infty)$. Thus, we denote the feature vector for a pair of particles as $\boldsymbol{x}_{a,b} \doteq [\boldsymbol{f}_a^\top \ \boldsymbol{f}_b^\top \ t]^\top \in \mathbb{R}^9$. These nine features were selected because they are computed efficiently and were expected to capture much of the relevant chemistry. We normalized each training, validation, and testing dataset so that the concatenation of the training and validation sets has a mean of 0 and a standard

deviation of 1 for each feature.

2.2.2 Loss functions

Each plasma simulation can be interpreted as a binomial distribution since each outcome is a Bernoulli trial for some probability p. We use the negative log-likelihood of the binomial distribution (B-NLL) as a loss function for a given simulation,

$$\ell_b(\hat{p}, m, n) \doteq -[m \log \hat{p} + (n-m)\log(1-\hat{p})] \,, \tag{1}$$

where $\hat{p}$ is the predicted probability, m is the number of events for the desired outcome (*e.g.*, sticking), n is the total number of events in a simulation.

We implement DeepSets with a sigmoidal activation $\sigma(u) \doteq (1+e^{-u})^{-1}$ on the output layer where $\sigma\colon \mathbb{R} \to [0,1]$ and directly optimize the B-NLL loss. Unfortunately, many ML libraries do not natively support binomial loss functions. In such cases, we can rewrite the binomial loss ℓ_b in terms of the logistic loss ℓ_l,

$$\begin{aligned} \ell_b(\hat{p}, m, n) &= -\left[m\ell_l(\hat{p}, 1) + (n-m)\ell_l(\hat{p}, 0)\right] \\ \text{for} \quad \ell_l(\hat{p}, b) &\doteq \begin{cases} \log \hat{p} & b = 1 \\ \log(1-\hat{p}) & b = 0 \end{cases} , \end{aligned} \tag{2}$$

where b indicates a class label. This can be interpreted as logistic regression that includes both classes for each set of trials but weights each "pseudosample" according to the number of positive and negative events. We use this approach for Logistic ElasticNet and LGBM.

Another perspective is to interpret the event probability as a scalar $p \in [0,1]$ and perform regression in logit-space. We cannot perform unconstrained regression directly on probabilities, as the model may predict unphysical values outside $[0,1]$. Instead, we apply the logistic unit (logit) $\sigma^{-1}(p) \doteq \log(1/(1-p))$ where $\sigma^{-1}\colon [0,1] \to \mathbb{R}$ to restrict the (untransformed) output range. Note that $\{0,1\}$ values cannot be predicted with a finite model output when using logits, so we clip the true probabilities at $[\epsilon, 1-\epsilon]$ for some small ϵ. We refer to this loss as the "Logit MSE" (L-MSE). To further emulate the binomial NLL, we weigh the loss for each simulation according to the number of trials and refer to it as the "Logit-Weighted MSE" (LW-MSE). The LW-MSE penalizes outliers more significantly than Binomial NLL, as seen in Figure 1. We use the LW-MSE for the unpenalized linear model, ElasticNet, KRR, and SVR, and also evaluate it on DeepSets and LGBM.

KNN does not utilize a loss function and is automatically restricted to the range $[0,1]$ as predictions are a weighted average of training data points, weighted by distance. We use a "naïve" predictor as a baseline, which predicts the mean of the training probabilities.

2.2.3 Permutation invariance

Because our two-particle systems are permutation invariant, we train permutation invariant models using either data manipulation or model construction.

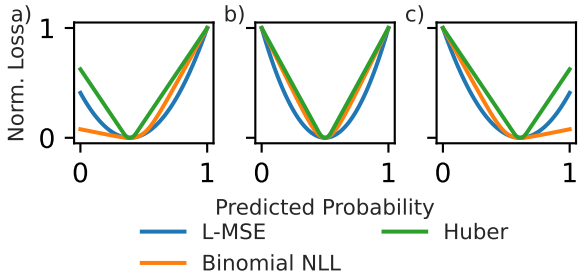

Figure 1: B-NLL, L-MSE, and logit-transformed Huber (L-H) losses rescaled and aligned for true P_{st}s of **a)** 0.1, **b)** 0.5, **c)** 0.9.

Some ML model implementations cannot be customized to be permutation invariant by construction, so we must adjust the dataset rather than the model itself. For linear models such as OLS and ElasticNet, permutation invariance can easily be achieved by defining $\bar{\boldsymbol{x}}_{a,b} \doteq [(\boldsymbol{f}_a^\top + \boldsymbol{f}_b^\top)/2 \;\; t]^\top \in \mathbb{R}^5$, which is equivalent to having equal model weights for the same indices of $\boldsymbol{f}_a, \boldsymbol{f}_b$. Crucially, this averaging approach is only valid for linear models and would reduce the expressiveness of nonlinear models such as LGBM. Instead, for LGBM, we augment the dataset to contain $\boldsymbol{x}_{a,b}$ and $\boldsymbol{x}_{b,a}$. Although helpful, this approach does not guarantee invariance, so we refer to it as pseudo-permutation invariant. In both cases, normalization is applied after transforming the training and validation feature vectors.

Other models can be directly modified to learn permutation invariant functions. KNN, KRR, and SVR all use distance metrics to express new points as combinations of training data points; KNN directly uses a distance metric to select and weight neighbors, and KRR and SVR use the RBF kernel

$$k(\boldsymbol{x}_{a,b}, \boldsymbol{x}_{c,d}) \doteq \exp\left(-\gamma d(\boldsymbol{x}_{a,b}, \boldsymbol{x}_{c,d})\right) \tag{3}$$

for some distance metric $d : \mathbb{R}^9 \times \mathbb{R}^9 \to [0, \infty)$ via the representor theorem. For all three methods, we use the permutation invariant distance metric

$$d(\boldsymbol{x}_{a,b}, \boldsymbol{x}_{c,d}) \doteq \min\left(\left\| \begin{bmatrix} \boldsymbol{f}_a - \boldsymbol{f}_c \\ \boldsymbol{f}_b - \boldsymbol{f}_d \\ t \end{bmatrix} \right\|_2^2, \left\| \begin{bmatrix} \boldsymbol{f}_a - \boldsymbol{f}_d \\ \boldsymbol{f}_b - \boldsymbol{f}_c \\ t \end{bmatrix} \right\|_2^2 \right) . \tag{4}$$

This value can be interpreted as the minimum distance across all particle permutations within $\boldsymbol{x}_{a,b}$ and $\boldsymbol{x}_{c,d}$.

Finally, we use the DeepSets [35] neural network (NN) architecture, which has the form

$$g\left(\begin{bmatrix} \boldsymbol{f}_a \\ \boldsymbol{f}_b \\ t \end{bmatrix}\right) \doteq \phi_\eta\left(\begin{bmatrix} \rho\left(\psi_\theta(\boldsymbol{f}_a), \psi_\theta(\boldsymbol{f}_b)\right) \\ t \end{bmatrix}\right) , \tag{5}$$

where $\phi_\eta \colon \mathbb{R}^4 \to \mathbb{R}^h, \psi_\theta : \mathbb{R}^{h+1} \to (0,1)$ are neural networks with a hidden width of h and parameters η, θ and $\rho : \mathbb{R}^{2h} \to \mathbb{R}^h$ is a feature-wise mean or max. This architecture is permutation invariant by construction and has been used for similar particle interaction problems [19]. Together, these approaches make our predictions (pseudo-)permutation invariant, which improves generalization capabilities.

2.2.4 Cross-validation

To estimate the models' performance under different scenarios, we considered multiple cross-validation (CV) techniques: 5-fold, leave-one-temperature-out, leave-one-impactor-out, and leave-one-cluster-out. Furthermore, we selected the model parameters using a grid search to reduce human bias. Notably, estimating model performance and performing model selection using the same cross-validation splits is known to overestimate performance and lead to biased models [37, 38]. To combat such bias, we estimated the model performance using "nested cross-validation." This approach estimates model performance using an outer CV loop, splits each training set into an inner CV loop, and uses the inner loop to select optimal hyperparameters for each outer fold. Specifically, for each outer fold, we select the parameters with the lowest average loss for the inner test datasets and report the loss of the outer test set using these parameters. This process, known as "nested CV" provides an almost unbiased estimate of the true error [37].

The inner CV loop was conducted similarly to the outer loop, with the outer training set being split for the inner CV loop. For the 5-fold CV, we perform the inner CV using another random 5-fold CV. We also perform leave-one-out CV on the inner loop for leave-one-temperature-and leave-one-cluster-out CV. However, because there were so many impactors, the inner fold was created by partitioning the training impactors into five folds, each containing multiple impactors. In short, the inner loop for parameter selection uses the same split criteria as the outer loop; our preliminary tests show that this approach improves generalization to unseen clusters and impactors.

We use a modified CV approach to provide train, validation, and test sets. The data is typically split into training and testing sets of relative size $k-1$ and 1. However, since the NN and LGBM utilize validation sets, we split our inner training dataset for these models into $k-2$ folds for training and one fold for validation. If a model (*e.g.*, linear regression) didn't use a validation set, we combined the training and validation sets.

An algorithmic representation is shown in the Supplemental Materials (SI Algorithms 1 and 2), while tables of the grid-searched parameters and values are included in Appendix Section C. We estimate the performance standard deviation by performing nested CV with five random seeds (random seeds were shared when splitting the dataset and initializing the model states). All other parameters were set to the library defaults.

3 Results and Discussion

3.1 Molecular Simulation

Previous work has analyzed the collisions of SiHx as a critical step for the growth of particles in silane plasma [15]. However, larger silanes also play a role in chemical growth despite decreasing concentration. Figure 2 shows P_{st}, the probability of a chemisorption or sticking event, for the collision of Si_2H_x with three of the simulated clusters.

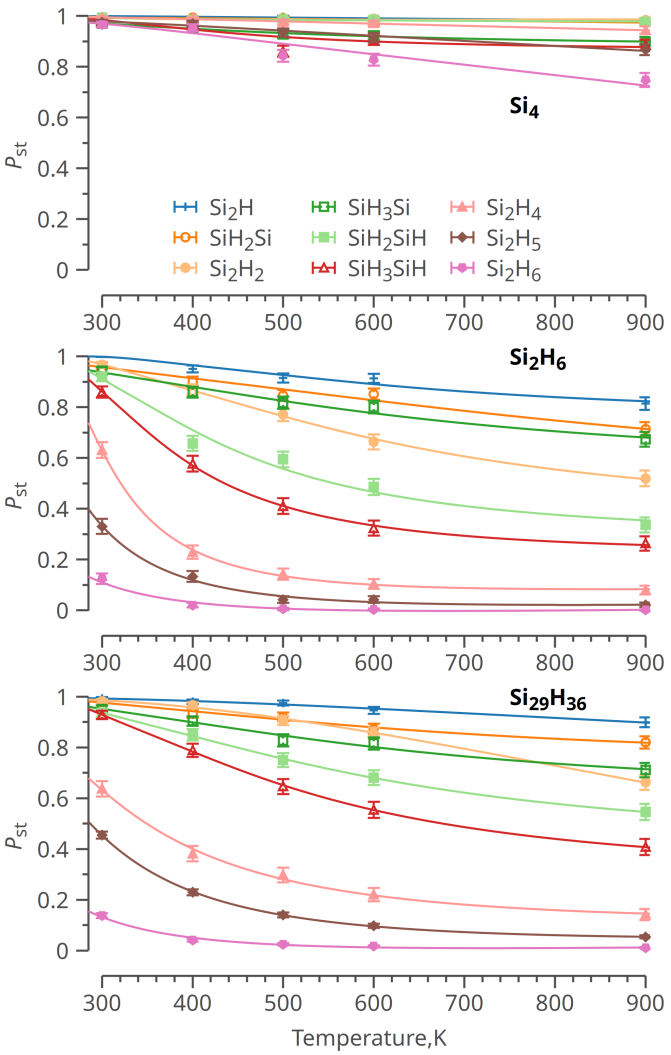

Figure 2: Temperature dependence of the sticking probability for collisions between different Si_2H_y and **a)** Si_4, **b)** Si_2H_6, and **c)** $Si_{29}H_{36}$. Lines show the fitted trend, described in (6). Error bars represent two standard deviations. Si_2H_y indicate molecules with balanced hydrogen distribution, while unbalanced fragments are expanded for clarity.

Similarly to previous work [15], the trends of P_{st} are

well approximated by

$$m(T, E, b, c) = (1 - c)f(T - b, E) + c \quad , \tag{6}$$

where T is the translational temperature, E is the kinetic energy, b and c are fitted constants, and f is the cumulative distribution function of the Maxwell-Boltzmann distribution:

$$f(T, E) = \text{erf}\left(\sqrt{\frac{E}{k_B T}}\right) - \frac{2}{\sqrt{\pi}}\sqrt{\frac{E}{k_B T}}\exp\left(-\frac{E}{k_B T}\right) \tag{7}$$

in which k_B is the Boltzmann constant and erf is the error function.

Compared to the results of SiH_y collisions, the P_{st} for Si_2H_y, while generally slightly higher, displays very similar trends. As expected, the sticking probability decreases at higher temperatures, with a stronger dependency observed for species with a lower number of radical electrons due to their high reactivity. As before, the number of unpaired radicals plays a crucial role in the reactivity. However, the picture is complicated by the effect of balanced *vs.* imbalanced hydrogens on the silane impactors. While a hydrogen imbalance results in greater reactivity, this effect is secondary to the overall hydrogen coverage. Outliers like Si_2H_2 do not follow the expected trend, displaying a lower-than-expected propensity to form bonds at higher temperatures.

Cluster size has a relatively small effect, mostly apparent when physisorption is relevant, whether as a step for the chemisorption or as a collision outcome. By analyzing the ratio between collisions that lead to chemisorption and the chemisorption and physisorption events (see Figure D1 in the Supplementary Material), we observe that physisorption plays an integral role in the kinetics for almost fully saturated species like $Si_2H_5$and Si_2H_4 isomers. These reactants, which are less reactive than the more unsaturated counterparts or require as much energy as the fully saturated silanes, are the most likely to control the kinetics of particle growth. It is worth noting that the lifetime of a physisorbed pair can vary from a few to tens of ps, even when the outcome is a chemisorption. As a result, in an experimental setting, other processes can occur in this timescale that the current model does not capture.

Finally, our simulations also study $Si_{29}H_x$ nanoparticles and hydrogen coverage to provide a quantitative relationship of the effect of hydrogen coverage of larger NPs on the sticking coefficients. In Figure 3, we compare sticking coefficients with coverages of 18, 27, 31, and 36 hydrogens. The same trends observed with other silane fragments are evident here: increasing hydrogen coverage while holding the temperature, impactor, and number of cluster silicon atoms monotonically increases the sticking probability.

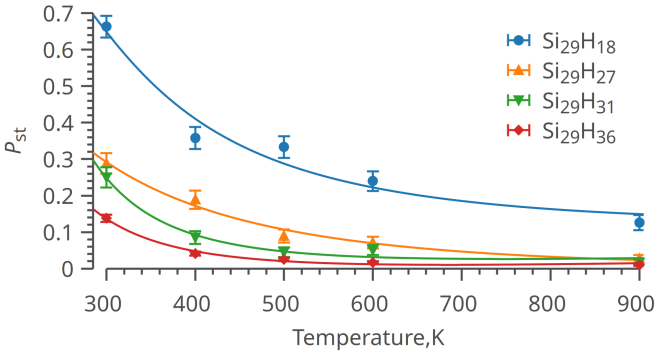

Figure 3: Sticking probability *vs.* temperature for the collisions between Si_2H_6 and Si_{29} cluster with different hydrogen coverages of $Si_{29}H_x$. The line represents the fit discussed in (6), and error bars (generally smaller than the symbols) represent two standard deviations.

3.2 Machine learning models

As described in the Methodology, we performed cross-validations using several splits to determine which environmental settings our models could and could not generalize. We test the overall capabilities of the model using a 5-fold CV (Figure 4) and out-of-distribution generalization by performing leave-one-temperature-, leave-one-impactor-, and leave-one-impactor-out CV (Figures 6 and 5, and SI Figure A1).

The binomial NLL depends on the number of trials in each simulation and is nonzero for perfect predictions. Furthermore, this nonzero floor is not constant and depends on the true probability. Thus, comparing the binomial NLL between folds may be misleading, as different numbers of trial runs or distributions of actual probabilities may dominate variations. To improve visualization, for plotting, we use the "adjusted B-NLL":

$$\ell_{\text{adj}}(\hat{p}, m, n) \doteq \frac{1}{n}\left(\ell_b\left(\hat{p}, m, n\right) - \ell_b\left(\frac{m}{n}, m, n\right)\right) \tag{8}$$

This loss weights all simulations equally, regardless of the number of trials run, and subtracts the NLL of a perfect prediction from the NLL of the actual prediction. For similar reasons, we plot the unweighted L-MSE instead of the LW-MSE. Unlike root mean squared error, these metrics do not correspond to intuitive notions of distance but still facilitate a quantitative performance comparison between methods. Thus, we also plot the true *vs.* predicted probability for the sticking event to provide an intuition of how individual models perform.

In the 5-fold CV setting (Figure 4), all models perform significantly better than the naïve model. Indeed, Figures 4 **c)** and **d)** show almost perfect agreement between true and predicted probabilities. Furthermore, we find that ML models can be highly robust to data subsampling (Figure 4 **e)**). Indeed, performance does not meaningfully decrease until less than 25% of the data is used for training, and good performance is achieved by training on only 15% of the data.

In leave-one-cluster-out testing (Figure 5), the models performed well for most unsampled $Si_{29}H_x$ clusters but

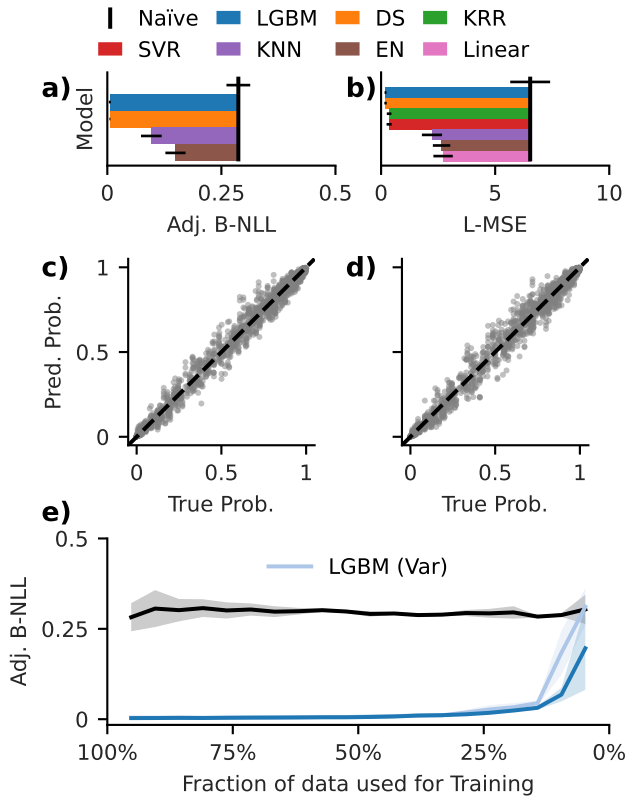

Figure 4: Performance of 5-fold cross-validation. **a)** and **b)** show the average performance of each model trained and evaluated using the adjusted B-NLL and L-MSE, respectively. Black bars indicate the standard deviation across all five folds and five random seeds. **c)** and **d)** show the LGBM predictions for all folds and seeds using the same loss functions as **a)** and **b)**, respectively. **e)** shows the adjusted B-NLL performance of the permutation invariant (dark blue) and variant (light blue) LGBM models and the naïve model as the fraction of training data decreases. The shaded region indicates the standard deviation across random seeds.

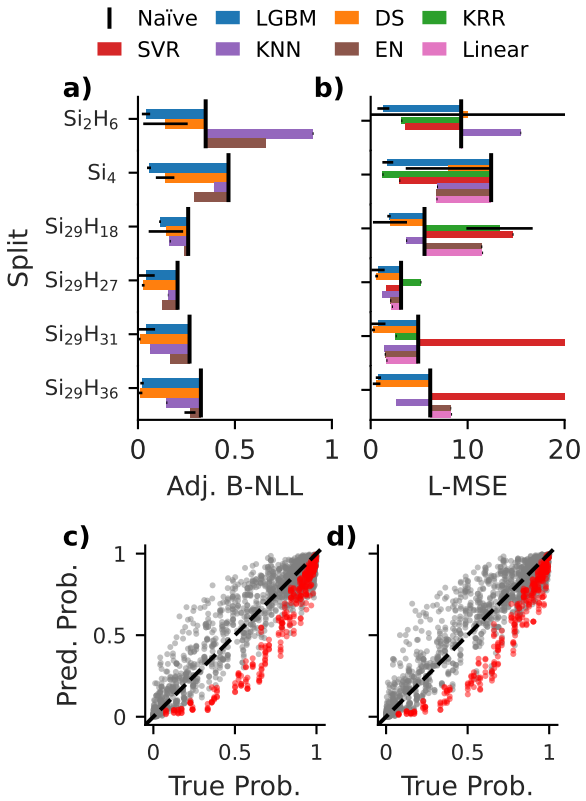

Figure 5: Performance of leave-one-cluster-out CV. **a)** and **b)** show the average performance of each model trained and evaluated using the adjusted B-NLL and L-MSE, respectively. Black bars indicate the standard deviation across random seeds. **c)** and **d)** show the predictions of LGBM for all folds and random seeds using the same loss functions as **a)** and **b)**, respectively. Predictions for $Si_{29}H_{18}$ are highlighted in red. The losses for SVR $Si_{29}H_{31}$ and $Si_{29}H_{36}$ in **b)** are 23.5 and 41.5 but are truncated for visualization purposes.

not for Si_2H_6, likely due to the presence only in the large cluster of low vibrational frequencies that can better accommodate the collision energy. Since our training set has four $Si_{29}H_x$ clusters and only one Si_2H_x and one Si_4H_x clusters, it appears that the model is biased towards the behavior of the $Si_{29}H_x$ clusters.

Notably, most models showed an increased error for $Si_{29}H_{18}$. The $Si_{29}H_x$ clusters start with the fully saturated $Si_{29}H_{36}$ molecule and become less saturated until we get to $Si_{29}H_{18}$. Therefore, we expect the error to be higher for $Si_{29}H_{36}$ because the model has only unsaturated $Si_{29}H_x$ clusters to train from, while for the remaining $Si_{29}H_x$ clusters, we expected similar errors. The slightly abnormal behavior of the predictions for collisions involving $Si_{29}H_{18}$ suggests interactions dominated by different reaction pathways, possibly related to H isomerization. The results for the impactors are similar (see Appendix Section A). Overall, the models are most effective when making predictions for similar-sized molecules, at least when provided with such a limited selection.

While the natural conclusion about the need for a wider variety of cluster sizes is correct, it should also be confronted with the fact that not all atomic arrangements are equally stable. The lower energy associated with specific structures (*e.g.*, spherical, truncated polyhedrons) may result in some clustering of the dominant reactive pathways, which may complicate even a model trained on a more varied dataset.

A similar analysis for temperature is shown in Figure 6, where we observe that the model performance is relatively consistent except at the lowest temperature. The naïve loss is highest for 300 K, and most models (except DeepSets) perform poorly for 300 K and even worse for 900 K. While we could not determine the reason for this difference beyond the difficulty of extrapolation compared to interpolation (which should affect the 900 K), it should

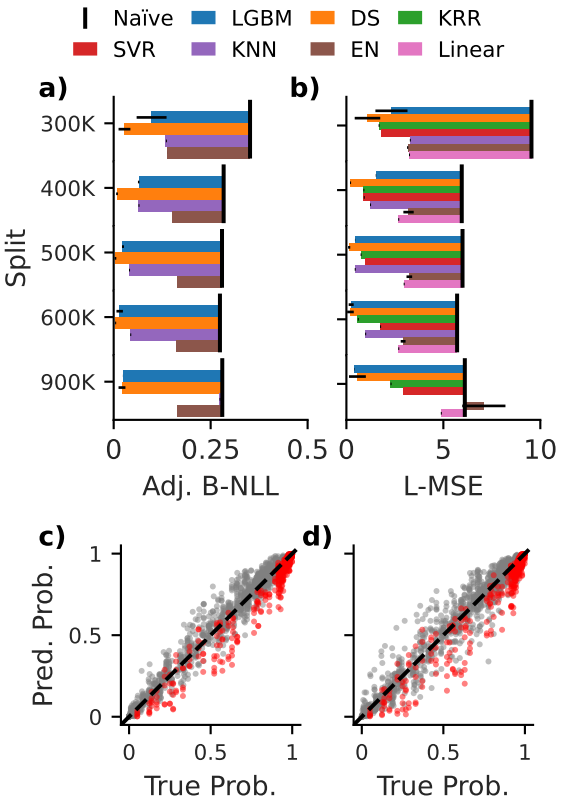

Figure 6: Performance of leave-one-temperature-out CV. **a)** and **b)** show the average performance of each model trained and evaluated using the adjusted B-NLL and L-MSE, respectively. **c)** and **d)** show the predictions of DeepSets for all folds and random seeds using the same loss functions as **a)** and **b)**, respectively. The predictions for 300 K are highlighted in red.

be noted that physisorption plays a much more significant role at this temperature, hinting again at the model sensitivity to underlying physical and chemical processes.

As shown in Figure 2 and SI Figure D1, P_{st} is nearly constant between 700 K and 900 K. Because LGBM learns piecewise-constant functions, it also performs constant extrapolations, which is ideal in this setting. However, P_{st} is hardly constant between 300 K and 400 K, meaning that constant extrapolations perform poorly. This is why other methods, such as DeepSets, outperform LGBM when extrapolating to lower temperatures and why LGBM outperforms all other methods but DeepSets when extrapolating to higher temperatures.

Notably, we find that permutation-variance significantly impacts the generalization capabilities of all models, as shown in Figure 5. This effect is particularly strong for Si_2H_6, Si_4, $Si_{29}H_{27}$. We suspect that this is partial because Si_4 and Si_2H_6 are more similar in size to the impactors than the other clusters. As a result, a permutation-variant model learns that the larger particles are usually on one side, which biases the predictions. Indeed, comparing Figures 5 and 7, we find that permutation-variant models achieve surprisingly high and low performance depending on the particle being held out, indicating a tendency to fit and an inability to generalize. Additionally, panels **c)** and **f)** in Figure 8 show that permutation-variant models make highly inconsistent predictions for each permutation of clusters and impactors. Indeed, predictions are inconsistent between permutations and are inaccurate unless the model is trained and tested on the same ordering of clusters and impactors. This variability demonstrates the importance of permutation-invariance for accurate modeling of P_{st} in nonthermal plasmas.

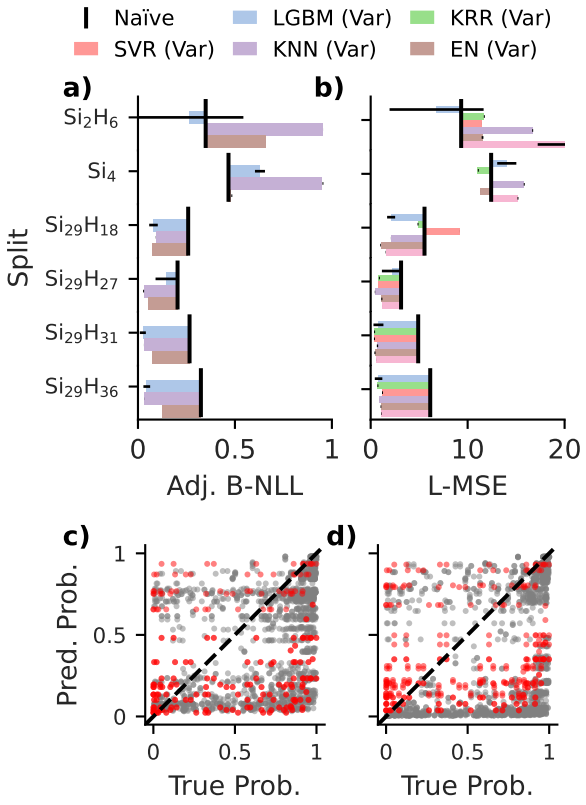

Figure 7: Performance of leave-one-cluster-out CV with permutation-variant models. **a)** and **b)** show the average performance of each model trained and evaluated using the adjusted B-NLL and L-MSE, respectively. Black bars indicate the standard deviation across random seeds. **c)** and **d)** show the LGBM (Var) predictions for each fold and random seed using the same loss functions as **a)** and **b)**, respectively. Here, we permute the particles before applying the model. Predictions for Si_2H_6 are highlighted in red.

Visual inspection of true *vs.* predicted probabilities shows that the B-NLL provides more robust predictions than the L-MSE. Both losses achieve good empirical performance, indicating that the L-MSE may be suitable when a binomial NLL loss cannot easily be added to a model. However, the B-NLL achieves more accurate predictions overall due to its less extreme penalization of out-

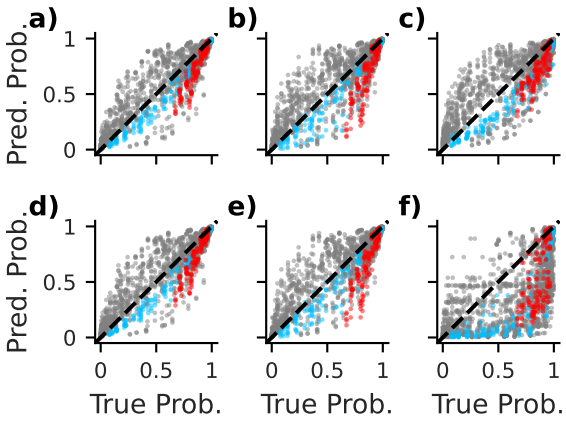

Figure 8: **a)**–**c)** predict P_{st} for cluster-impactor pairs, while **d)**–**f)** predict P_{st} for impactor-cluster pairs. **a)**, **b)**, **d)**, and **e)** show pseudo-permutation invariant LGBM, while **c)** and **f)** show standard LGBM. **a)**, **c)**, **d)**, and **f)** are trained with the Binomial NLL, while **b)** and **e)** are trained using the LW-MSE. Predictions for SiH_3−Si and Si_2H_4 are highlighted in blue and red, respectively. All random seeds are shown.

liers. For example, compare the predictions for LGBM trained with each loss on leave-one-impactor-out cross-validation: Although the worst-performing impactor is equally bad for both B-NLL and L-MSE, the error is lower for intermediately-performing impactors such as SiH_3−SiH and Si_2H_4 when considering the B-NLL (SI Figure A1). This is reminiscent of robust regression, where robust losses (*e.g.*, Huber loss) are chosen because they do not over-emphasize outliers as significantly as the squared error. Indeed, the B-NLL loss sits between the L-MSE and logit-transformed Huber Loss (L-H) in Figure 1. Thus, we recommend using the Binomial NLL when possible, and we suspect that further improvements are possible using robust binomial or logistic regression approaches [39].

We find that, in nearly all settings, ML models significantly outperform the naïve prediction. However, we note that models generally perform best when interpolating (rather than extrapolating) for temperature and that better performance may be achieved at higher temperatures. Additionally, DeepSets excels at extrapolating to unseen temperatures, and LGBM is the most consistent at successfully extrapolating to unseen structures. Finally, the loss function and permutation invariance can significantly affect model performance and generalization. This result suggests that, given a correctly chosen model architecture, we can focus our simulations on a small subset of important conditions (*e.g.*, edge temperatures) to derive P_{st} across many diverse settings effectively.

4 Conclusions

Particle growth in a high-energy gas phase, such as during combustion or in nonthermal plasma, is a complex, non-linear process that requires accurate modeling. Even when narrowing the scope to a specific system and set of reactions, capturing the rates and mechanism remains computationally challenging, even using classical approximations. While such detailed descriptions are not always necessary, a more nuanced description of the reaction rates can benefit several contexts, such as more accurate reaction rates, hyper-doping, and core-shell nanoparticle production. The subtle differences between the various Si_2H_y species simulated in this paper, as well as SiH_y, are symptomatic of a series of competing phenomena (*e.g.*, physisorption, energy redistribution) that cannot be easily generalized and that can lead to systematic biases when ignored.

To address this, we have focused on training and testing several permutation-invariant ML models to reduce the computational effort associated with these simulations. Our results show that nearly 90% of interactions can be predicted using machine learning without significantly impacting accuracy. Furthermore, we have demonstrated the importance of principled loss functions, model architectures, and sampling procedures for deriving accurate and reliable predictions. Figure 4 shows that, in general, simple system-specific features are descriptive enough to predict sticking probabilities for silane nanoparticles in nonthermal plasma after training on only a fraction of simulated interactions. Additionally, Figure 6 shows that our model can extrapolate and interpolate quite well for unsampled temperatures. However, Figure 5 indicates our specific combination of ML models and input features has difficulty extrapolating to nanoparticles with different degrees of saturation.

These findings demonstrate that ML methods can significantly reduce the computational cost of computing the results of complex reactions in nonthermal plasma and other difficult-to-model systems. However, careful selection of model architecture and training data is crucial to ensure the generalizability of predictions. Based on our results, we conclude that the most effective acceleration method involves simulating a subset of particles across a range of temperatures (especially the upper and lower bounds of the expected range), maintaining a balance of relevant molecular properties (*e.g.*, H-saturation in this case) in the training set, and then training a permutation-invariant ML model using the binomial negative log-likelihood.

While this study focused on the data collected for a specific system, namely the sticking probability of silanes computed through classical reactive molecular dynamics, the overall approach, both molecular dynamics simulations and analysis of ML models, is relatively general. As such, we expect that similar methods can be readily adapted to generate more computationally efficient and

realistic growth parameters for NTPs, thus improving the efficiency and accuracy of simulations.

5 Reproducibilty

Supporting data and code are currently available through this link: `https://gitlab.eecs.umich.edu/mattrmd-public/ntp-silicon`. Data will be provided via a DOI-minting repository upon acceptance.

6 CRediT authorship contribution statement

Matt Raymond: Formal analysis, investigation, methodology, software, supervision, validation, visualization, writing - original draft, writing - review & editing. **Paolo Elvati:** Conceptualization, data curation, investigation, methodology, software, supervision, visualization, writing - original draft, writing - review & editing. **Jacob Saldinger:** Conceptualization, data curation, software, supervision, writing - original draft. **Jonathan Lin:** Formal analysis, investigation, software, visualization, writing - original draft. **Xuetao Shi:** Data curation, software. **Angela Violi:** Conceptualization, funding acquisition, project administration, resources, supervision, writing - review & editing.

7 Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has been supported by the US Army Research Office MURI Grant No. W911NF-18-1-0240 and by the NSF ECO-CBET No. F059554.

References

[1] Mangolini L and Kortshagen U 2009 *Phys. Rev. E* **79**(2) 026405 URL https://doi.org/10.1103/PhysRevE.79.026405

[2] Gao X, Cui Y, Levenson R M, Chung L W K and Nie S 2004 *Nature Biotechnology* **22** 969–976 ISSN 1546-1696 URL https://doi.org/10.1038/nbt994

[3] Fujioka K, Hiruoka M, Sato K, Manabe N, Miyasaka R, Hanada S, Hoshino A, Tilley R D, Manome Y, Hirakuri K and Yamamoto K 2008 *Nanotechnology* **19** 415102 URL https://doi.org/10.1088/0957-4484/19/41/415102

[4] Moore D, Krishnamurthy S, Chao Y, Wang Q, Brabazon D and McNally P J 2011 *physica status solidi (a)* **208** 604–607 URL https://doi.org/10.1002/pssa.201000381

[5] Saadane O, Longeaud C, Lebib S and Roca i Cabarrocas P 2003 *Thin Solid Films* **427** 241–246 ISSN 0040-6090 proceedings of Symposium K on Thin Film Materials for Large Area Electronics of the European Materials Research Society (E-MRS) 2002 Spring Conference URL https://www.sciencedirect.com/science/article/pii/S0040609002011938

[6] Doğan I and van de Sanden M C M 2016 *Plasma Processes and Polymers* **13** 19–53 URL https://doi.org/10.1002/ppap.201500197

[7] Weis S, Körmer R, Jank M P M, Lemberger M, Otto M, Ryssel H, Peukert W and Frey L 2011 *Small* **7** 2853–2857 URL https://doi.org/10.1002/smll.201100703

[8] Astruc D, Lu F and Aranzaes J R 2005 *Angewandte Chemie International Edition* **44** 7852–7872 URL https://doi.org/10.1002/anie.200500766

[9] Boufendi L and Bouchoule A 1994 *Plasma Sources Science and Technology* **3** 262 URL https://doi.org/10.1088/0963-0252/3/3/004

[10] Lanham S J, Polito J, Xiong Z, Kortshagen U R and Kushner M J 2022 *Journal of Applied Physics* **132** 073301 URL https://doi.org/10.1063/5.0100380

[11] Krüger F, Gergs T and Trieschmann J 2019 *Plasma Sources Science and Technology* **28** 035002 URL https://doi.org/10.1088/1361-6595/ab0246

[12] Kortshagen U and Bhandarkar U 1999 *Phys. Rev. E* **60**(1) 887–898 URL https://doi.org/10.1103/PhysRevE.60.887

[13] Agarwal P and Girshick S L 2014 *Plasma Chemistry and Plasma Processing* **34** 489–503 URL https://doi.org/10.1007/s11090-013-9511-3

[14] Le Picard R, Markosyan A H, Porter D H, Girshick S L and Kushner M J 2016 *Plasma Chemistry and Plasma Processing* **36** 941–972 ISSN 1572-8986 URL https://doi.org/10.1007/s11090-016-9721-6

[15] Shi X, Elvati P and Violi A 2021 *Journal of Physics D: Applied Physics* **54** 365203

[16] Lanham S J, Polito J, Shi X, Elvati P, Violi A and Kushner M J 2021 *Journal of Applied Physics* **130** ISSN 0021-8979 URL https://doi.org/10.1063/5.0062255

[17] Husmann E, Polito J, Lanham S, Kushner M J and Thimsen E 2023 *Plasma Chemistry and Plasma Processing* **43** 225–245 URL https://doi.org/10.1007/s11090-022-10299-3

[18] Bal K M and Neyts E C 2021 *Journal of Physics D: Applied Physics* **54** 394004 URL https://doi.org/10.1088/1361-6463/ac113a

[19] Saldinger J C, Raymond M, Elvati P and Violi A 2023 *Proceedings of the Combustion Institute* URL https://doi.org/10.1016/j.proci.2022.08.109

[20] Gidon D, Pei X, Bonzanini A D, Graves D B and Mesbah A 2019 *IEEE Transactions on Radiation and Plasma Medical Sciences* **3** 597–605 URL https://doi.org/10.1109/TRPMS.2019.2910220

[21] van der Gaag T, Onishi H and Akatsuka H 2021 *Physics of Plasmas* **28** 033511 ISSN 1070-664X URL https://doi.org/10.1063/5.0023928

[22] Liang C, Huang D, Lu S and Feng Y 2023 *Phys. Rev. Res.* **5**(3) 033086 URL https://doi.org/10.1103/PhysRevResearch.5.033086

[23] Han S, Ceiler M, Bidstrup S, Kohl P and May G 1994 *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part A* **17** 174–182 URL https://doi.org/10.1109/95.296398

[24] Guessasma S, Montavon G and Coddet C 2004 *Computational Materials Science* **29** 315–333 ISSN 0927-0256 URL https://doi.org/10.1016/j.commatsci.2003.10.007

[25] Pakseresht A H, Ghasali E, Nejati M, Shirvanimoghaddam K, Javadi A H and Teimouri R 2015 *The International Journal of Advanced Manufacturing Technology* **76** 1031–1045 ISSN 1433-3015 URL https://doi.org/10.1007/s00170-014-6212-x

[26] Gergs T, Mussenbrock T and Trieschmann J 2023 *Journal of Physics D: Applied Physics* **56** 194001 URL https://doi.org/10.1088/1361-6463/acc07e

[27] Hong S, May G and Park D C 2003 *IEEE Transactions on Semiconductor Manufacturing* **16** 598–608 URL https://doi.org/10.1109/TSM.2003.818976

[28] Kim B and May G 1994 *IEEE Transactions on Semiconductor Manufacturing* **7** 12–21 URL https://doi.org/10.1109/66.286829

[29] Kwon O, Lee N and Kim K 2022 *IEEE Transactions on Semiconductor Manufacturing* **35** 256–265 URL https://doi.org/10.1109/TSM.2022.3154366

[30] Abd Jelil R, Zeng X, Koehl L and Perwuelz A 2013 *Engineering Applications of Artificial Intelligence* **26** 1854–1864 ISSN 0952-1976 URL https://doi.org/10.1016/j.engappai.2013.03.015

[31] Plimpton S 1995 *Journal of Computational Physics* **117** 1–19 URL https://doi.org/10.1006/jcph.1995.1039

[32] Rappe A K and Goddard W A 1991 *The Journal of Physical Chemistry* **95** 3358–3363

[33] Nakano A 1997 *Computer Physics Communications* **104** 59–69 URL https://doi.org/10/cmbzsw

[34] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M and Duchesnay E 2011 *Journal of Machine Learning Research* **12** 2825–2830 version: 1.5.1 URL https://dl.acm.org/doi/10.5555/1953048.2078195

[35] Zaheer M, Kottur S, Ravanbakhsh S, Poczos B, Salakhutdinov R R and Smola A J 2017 Deep sets *Advances in Neural Information Processing Systems* vol 30 ed Guyon I, Luxburg U V, Bengio S, Wallach H, Fergus R, Vishwanathan S and Garnett R (Curran Associates, Inc.) URL https://papers.nips.cc/paper_files/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html

[36] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q and Liu T Y 2017 *Advances in neural information processing systems* **30** 3146–3154 URL https://papers.nips.cc/paper_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

[37] Varma S and Simon R 2006 *BMC Bioinformatics* **7** 91 ISSN 1471-2105 URL https://doi.org/10.1186/1471-2105-7-91

[38] Cawley G C and Talbot N L C 2010 *Journal of Machine Learning Research* **11** 2079–2107 URL http://jmlr.org/papers/v11/cawley10a.html

[39] Feng J, Xu H, Mannor S and Yan S 2014 Robust logistic regression and classification *Advances in Neural Information Processing Systems* vol 27 ed Ghahramani Z, Welling M, Cortes C, Lawrence N and Weinberger K (Curran Associates, Inc.) URL https://papers.nips.cc/paper/2014/hash/6cdd60ea0045eb7a6ec44c54d29ed402-Abstract.html

A Leave Impactor Out

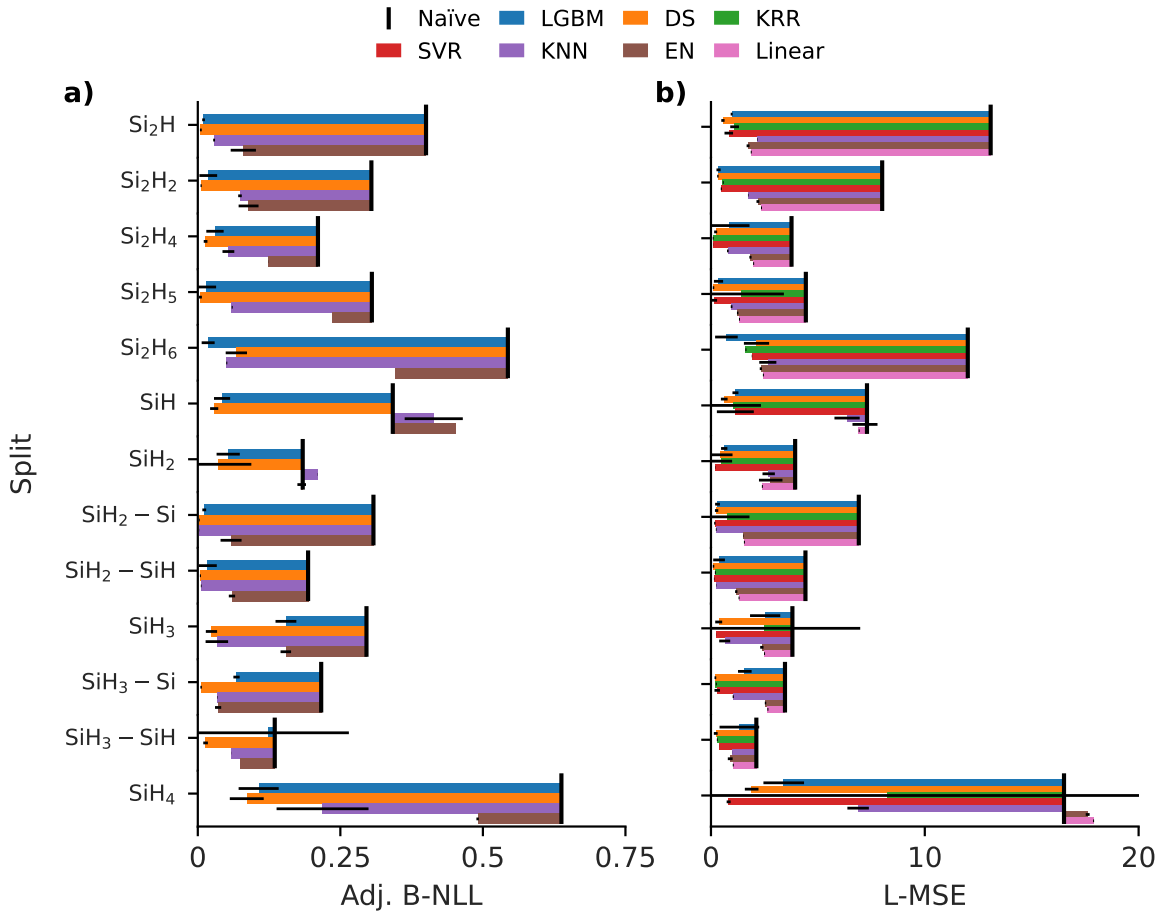

Figure A1: Performance of leave-one-impactor-out CV. **a)** and **b)** show the average performance of each model trained and evaluated using the adjusted B-NLL and L-MSE, respectively. Black bars indicate the standard deviation across random seeds.

For leave-one-impactor-out (SI Figure A1), some models perform especially poorly for SiH, SiH_2, SiH_3−Si, and SiH_3−SiH. There are several potential explanations for this. One is that the naïve prediction error for these particles is already significantly lower than the other particles, leaving less room for improvement. For a couple of impactors, namely SiH_4 and SiH_3−SiH_3, most models had nearly double the error as they did for other impactors. This may be caused by overfitting due to the large amount of remaining data for partially saturated molecules, suggesting that optimal training data would contain a higher percentage of fully saturated molecules. Alternatively, it could simply mean that the sticking probabilities for this molecule are outliers. Regardless, DeepSets (and LGBM) significantly outperform the naïve model in all (most) cases.

B CV Method

In Supplementary Algorithms 1 and 2, we include the details of the cross-validation methods used in this work.

Input : Dataset D, parameter grid $\mathcal{G}$, parametric model $f_{\theta,g}$
Output: Performance metric for each outer testing set
`// Arrays holding outer test metric`
$\mu \leftarrow []$;
foreach $g \in \mathcal{G}$ **do**
  foreach $k \in [5]$ **do**
    $k' \leftarrow (k+1) \mod 5$;
    `// Outer cross-validation loop`
    $D^{\text{outer}}_{\text{train}} \leftarrow D$ split into 5 folds with the k and k'-th folds removed;
    $D^{\text{outer}}_{\text{test}} \leftarrow k$-th fold of D;
    $D^{\text{outer}}_{\text{val}} \leftarrow k'$-th fold of D;
    `// Array of losses`
    $\eta_g \leftarrow []$;
    foreach $j \in [5]$ **do**
      `// Inner cross-validation loop`
      $D^{\text{inner}}_{\text{train}} \leftarrow D^{\text{outer}}_{\text{train}}$ split into 5 folds with the j and j'-th folds removed;
      $D^{\text{inner}}_{\text{test}} \leftarrow j$-th fold of $D^{\text{outer}}_{\text{train}}$;
      $D^{\text{inner}}_{\text{val}} \leftarrow j'$-th fold of $D^{\text{outer}}_{\text{train}}$;
      $\theta^* \leftarrow \arg\min_\theta f_{\theta,g}$ with training and validation sets $D^{\text{inner}}_{\text{train}}$ and $D^{\text{inner}}_{\text{train}}$;
      Append the loss computed on $D^{\text{inner}}_{\text{test}}$ to η_g;
    end
    `// Select the optimal parameters`
    $g^* \leftarrow \arg\min_g \sum \eta_g/5$;
    `// Use these parameters to retrain the model`
    $\theta^* \leftarrow \arg\min_\theta f_{\theta,g^*}$ with training and validation sets $D^{\text{outer}}_{\text{train}}$ and $D^{\text{outer}}_{\text{train}}$;
    Append the loss computed on $D^{\text{outer}}_{\text{test}}$ to μ;
  end
end
`// Return array of performance metrics on the test sets`
return μ

Algorithm B1: Grid search with nested 5-fold cross-validation for one seed.

Input : Dataset D, parameter grid $\mathcal{G}$, set of groups $\mathcal{C}$, parametric model $f_{\theta,g}$
Output: Performance metric for each outer testing set
`// Arrays holding outer test metric`
$\mu \leftarrow []$;
foreach $g, \in \mathcal{G}$ **do**
  foreach $k \in [|\mathcal{C}|]$ **do**
    $k' \leftarrow (k+1) \mod |\mathcal{C}|$;
    `// Outer cross-validation loop`
    $D^{\text{outer}}_{\text{train}} \leftarrow D$ split into groups from $\mathcal{C}$ with the k and k'-th groups removed;
    $D^{\text{outer}}_{\text{test}} \leftarrow k$-th group of D;
    $D^{\text{outer}}_{\text{val}} \leftarrow k'$-th group of D;
    `// Array of losses`
    $\eta_g \leftarrow []$;
    foreach $j \in [|\mathcal{C} \setminus \{k, k'\}|]$ **do**
      `// Inner cross-validation loop`
      $D^{\text{inner}}_{\text{train}} \leftarrow D^{\text{outer}}_{\text{train}}$ split into groups from $\mathcal{C} \setminus \{k, k'\}$ with the j and j'-th groups removed;
      $D^{\text{inner}}_{\text{test}} \leftarrow j$-th group of $D^{\text{outer}}_{\text{train}}$;
      $D^{\text{inner}}_{\text{val}} \leftarrow j'$-th group of $D^{\text{outer}}_{\text{train}}$;
      $\theta^* \leftarrow \arg\min_\theta f_{\theta,g}$ with training and validation sets $D^{\text{inner}}_{\text{train}}$ and $D^{\text{inner}}_{\text{train}}$;
      Append the loss computed on $D^{\text{inner}}_{\text{test}}$ to η_g;
    end
    `// Select the optimal parameters`
    $g^* \leftarrow \arg\min_g \sum \eta_g / |\mathcal{C} \setminus \{k, k'\}|$;
    `// Use these parameters to retrain the model`
    $\theta^* \leftarrow \arg\min_\theta f_{\theta,g^*}$ with training and validation sets $D^{\text{outer}}_{\text{train}}$ and $D^{\text{outer}}_{\text{train}}$;
    Append the loss computed on $D^{\text{outer}}_{\text{test}}$ to μ;
  end
end
`// Return array of performance metrics on the test sets`
return μ

Algorithm B2: Grid search with nested leave-x-out cross-validation for one seed. In our setting, a "group" may be a cluster, impactor, or temperature. When $|\mathcal{C}|$ is large, we instead let j, j' be sets of groups instead of individual groups to reduce runtime.

C Grid Search

Parameters used for model selection in the inner 5-fold cross-validation described in the Methods, Section 3.2.

Parameter Name	**Values**
Epochs	100 000
Early stopping epochs	1 000
Activation function	`relu`
Batch size	64
Width	$\{64, 128\}$
Depth of each subnetwork	$\{2, 3\}$
Learning rate	0.001
Aggregation function ρ	$\{$`mean`, `max`$\}$
Optimizer	`adam`

Table C1: Grid search parameters for **DeepSets**

Parameter Name	**Values**
α (ℓ_1, ℓ_2 penalty weight)	$\{0.0001, 0.001, 0.01, 0.1, 1, 10\}$
ℓ_1 ratio	$\{0, 0.25, 0.5, 0.75, 1\}$
Maximum iterations	10 000 000

Table C2: Grid search parameters for **ElasticNet** (LW-MSE)

Parameter Name	**Values**
C (regularization)	$\{0.1, 1, 10, 100, 1\,000\}$
ε (tube)	$\{0.0001, 0.001, 0.01, 0.1, 1\}$
γ (scale for RBF kernel)	$\{0.01, 0.1, 1, 10\}$

Table C3: Grid search parameters for **SVR**

Parameter Name	**Values**
α (regularization)	$\{0.01, 0.1, 1, 10, 100\}$
γ (scale for RBF kernel)	$\{0.01, 0.1, 1, 10, 100\}$

Table C4: Grid search parameters for **KRR**

Parameter Name	**Values**
Neighbors	$\{3, 6, 9\}$
Weighting method	$\{$`uniform`, `distance`$\}$
p (for the ℓ_p-norm distance)	$\{1, 2\}$

Table C5: Grid search parameters for **KNN**

Parameter Name	**Values**
Number of estimators	$\{100, 1\,000\}$
alpha (ℓ_1 regularization)	$\{0, 0.1, 1\}$
λ (ℓ_2 regularization)	$\{0, 0.1, 1.0\}$

Table C6: Grid search parameters for **LGBM**

Parameter Name	Values
C (inverse of regularization strength)	$\{0.01, 0.1, 1.0\}$
ℓ_1 ratio	$\{0, 0.25, 0.5, 0.75, 1\}$
Solver	`saga`
Penalty	`elasticnet`

Table C7: Grid search parameters for **ElasticNet** (Binomial loss via Logistic Regression)

D MD additional results

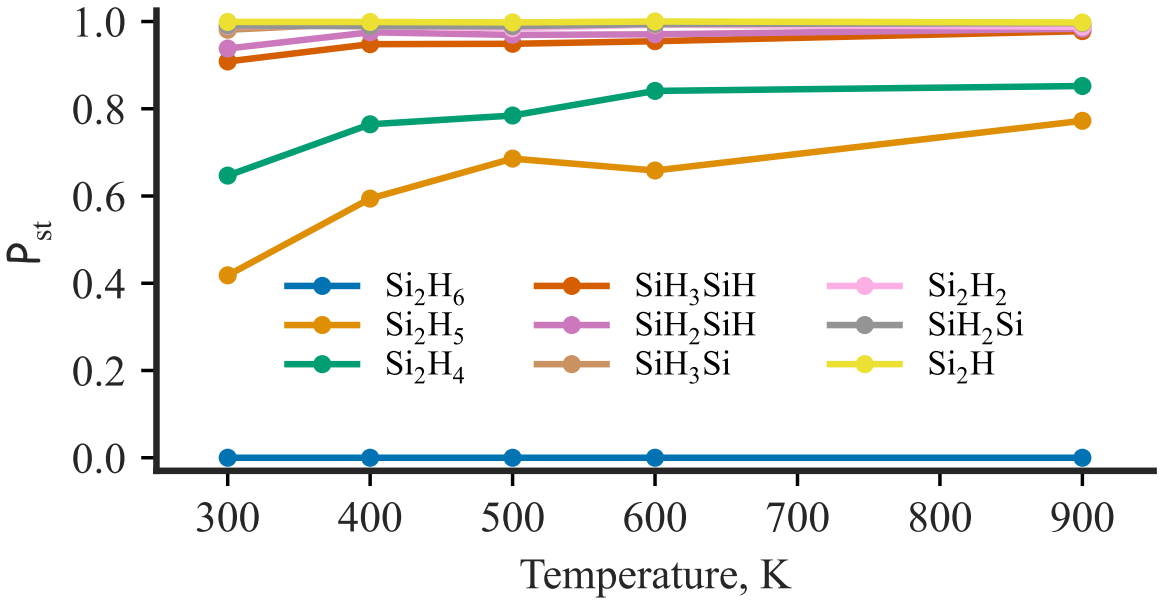

Figure D1: Fraction of chemisorption sticking events between $Si_{29}H_{36}$ and different Si_2H_x silicon fragments at various temperatures.

E True *vs.* Predicted values

Here, we plot each cross-validation procedure's true *vs.* predicted sticking probability. We plot the test points for every cross-validation loop that used the selected parameters.

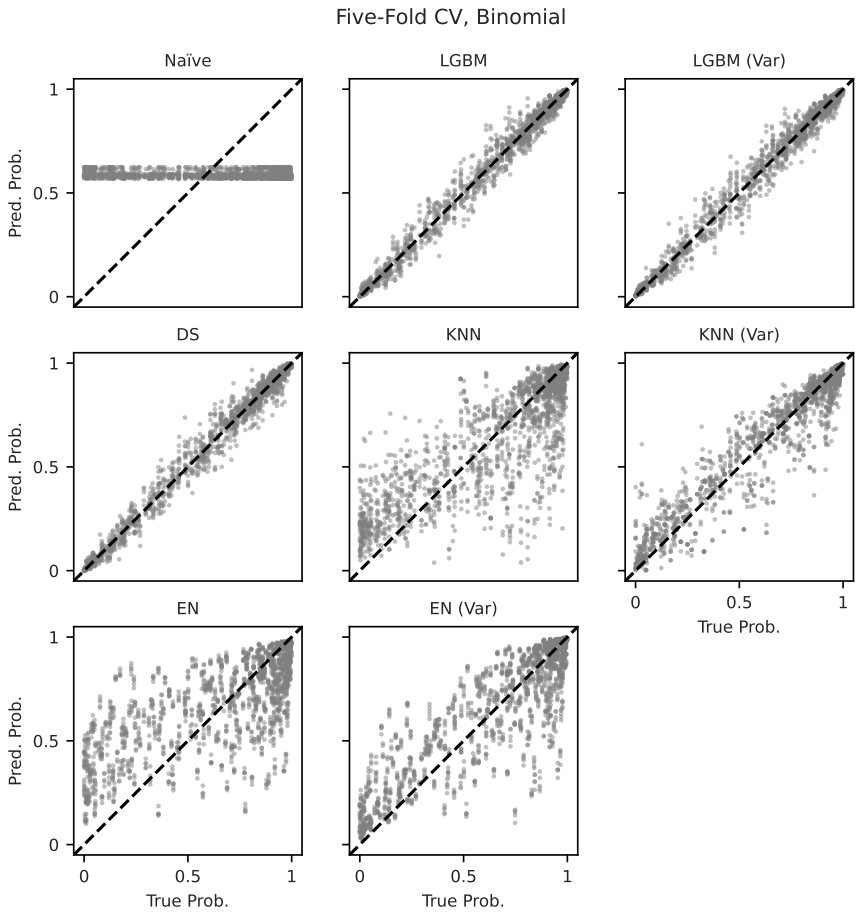

Figure E1: True *vs.* predicted probabilities for each model using nested **five-fold** cross-validation and the Binomial loss.

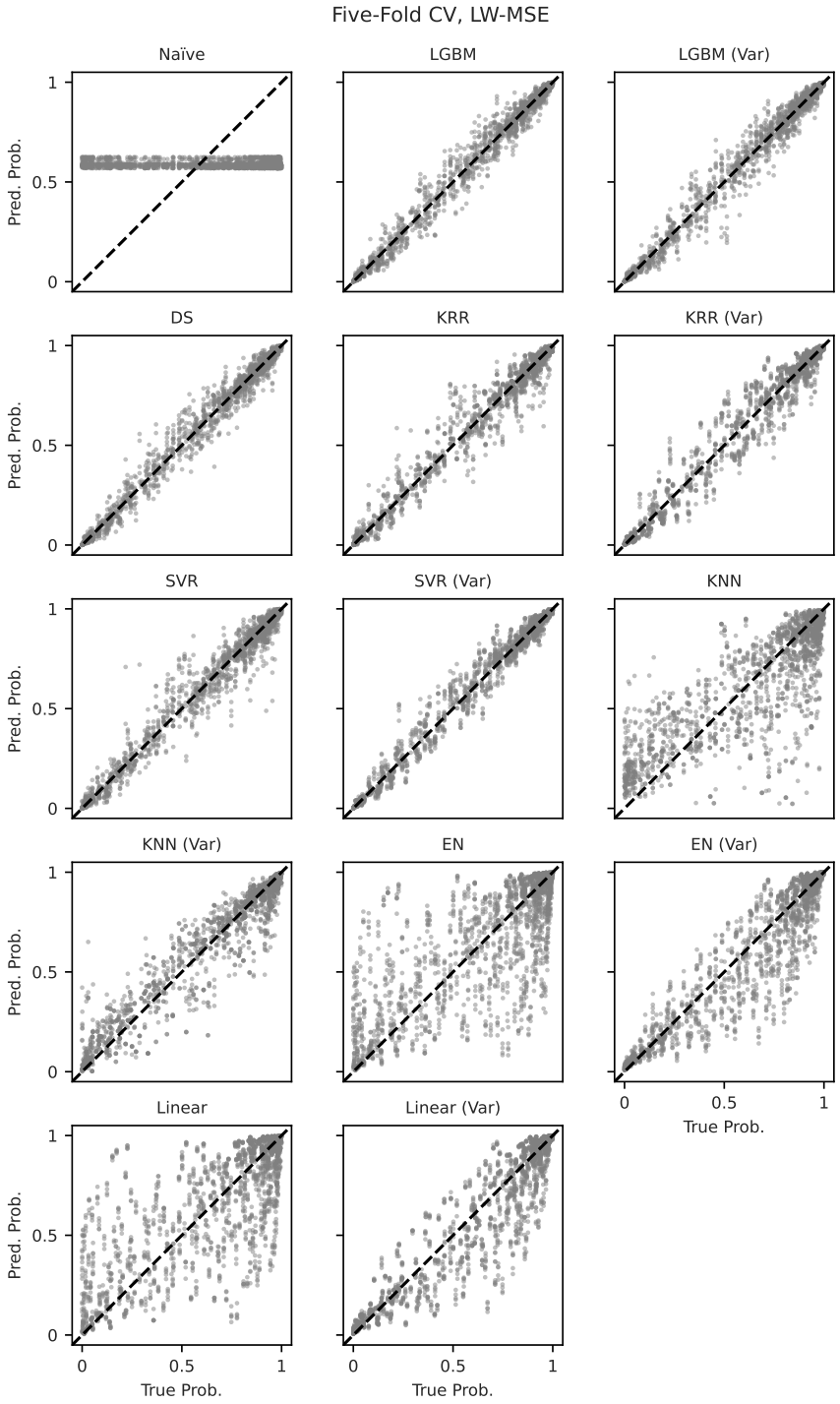

Figure E2: True *vs.* predicted probabilities for each model using nested **five-fold** cross-validation and the LW-MSE loss.

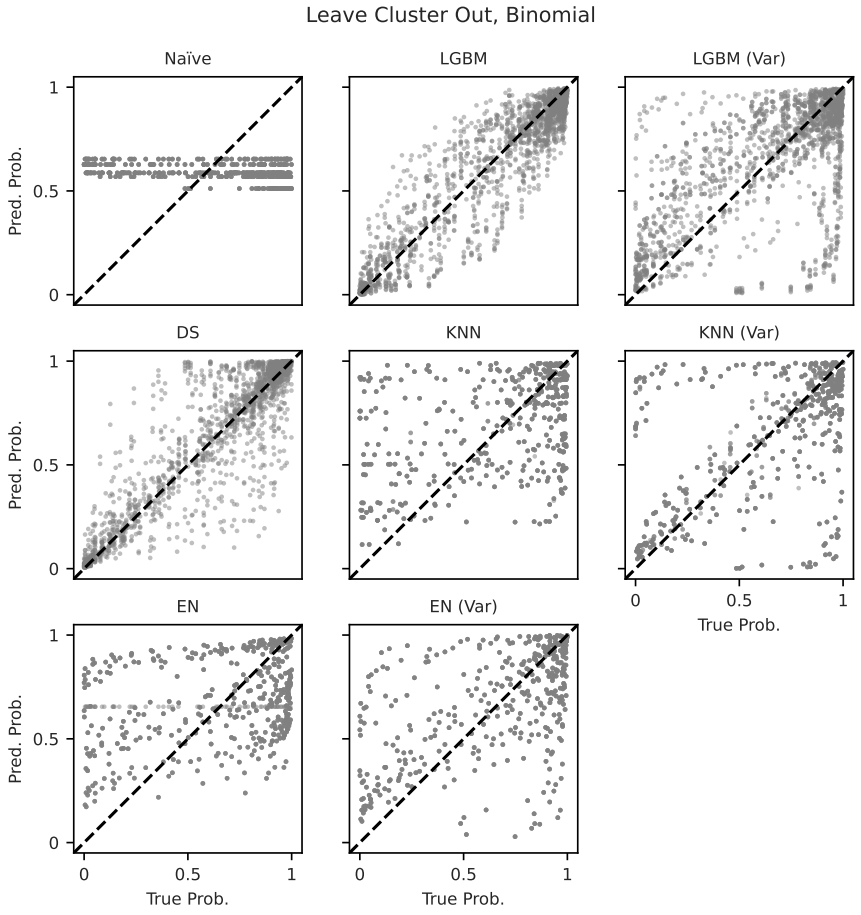

Figure E3: True *vs.* predicted probabilities for each model using nested **leave-one-cluster-out** cross-validation and the Binomial loss.

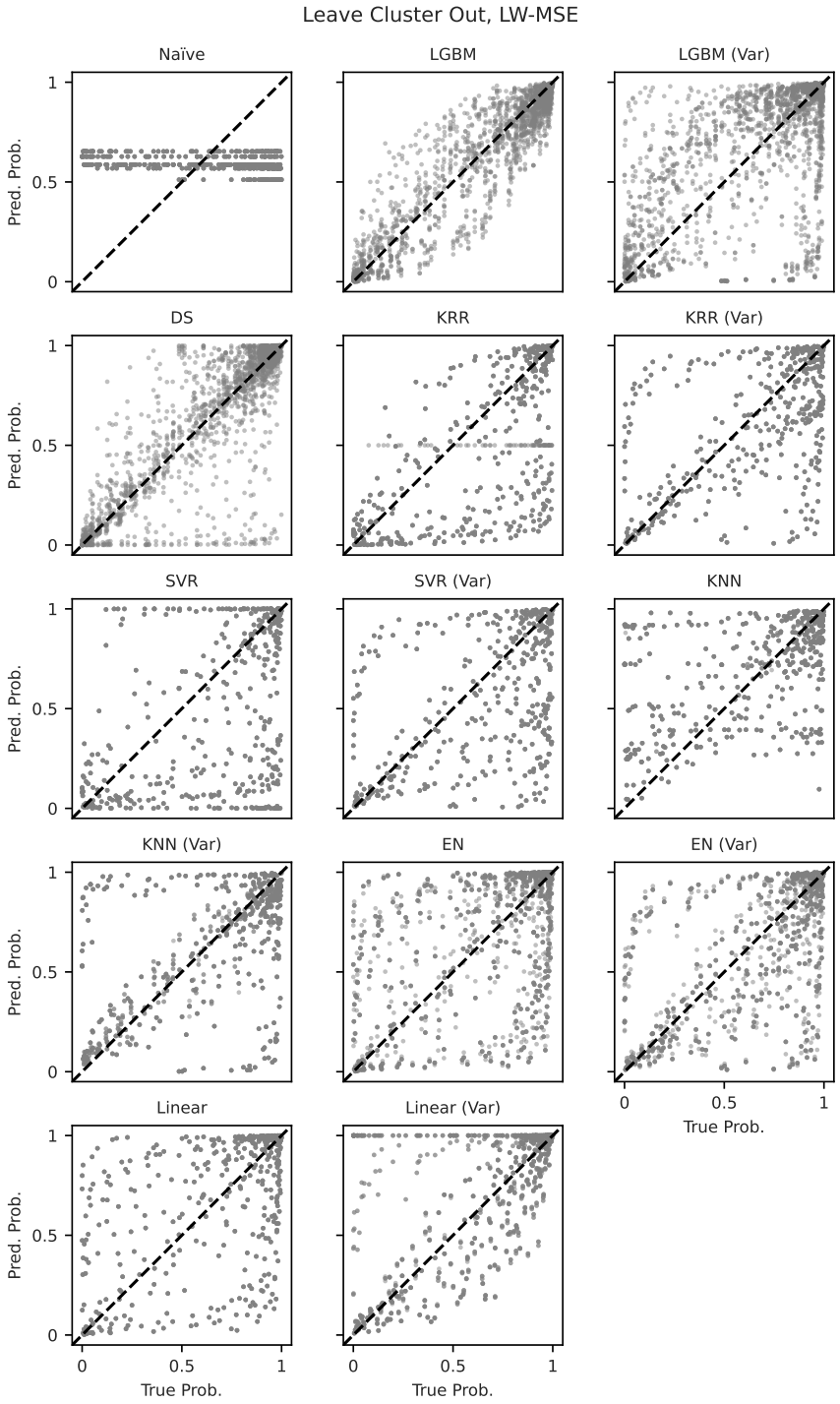

Figure E4: True *vs.* predicted probabilities for each model using nested **leave-one-cluster-out** cross-validation and the LW-MSE loss.

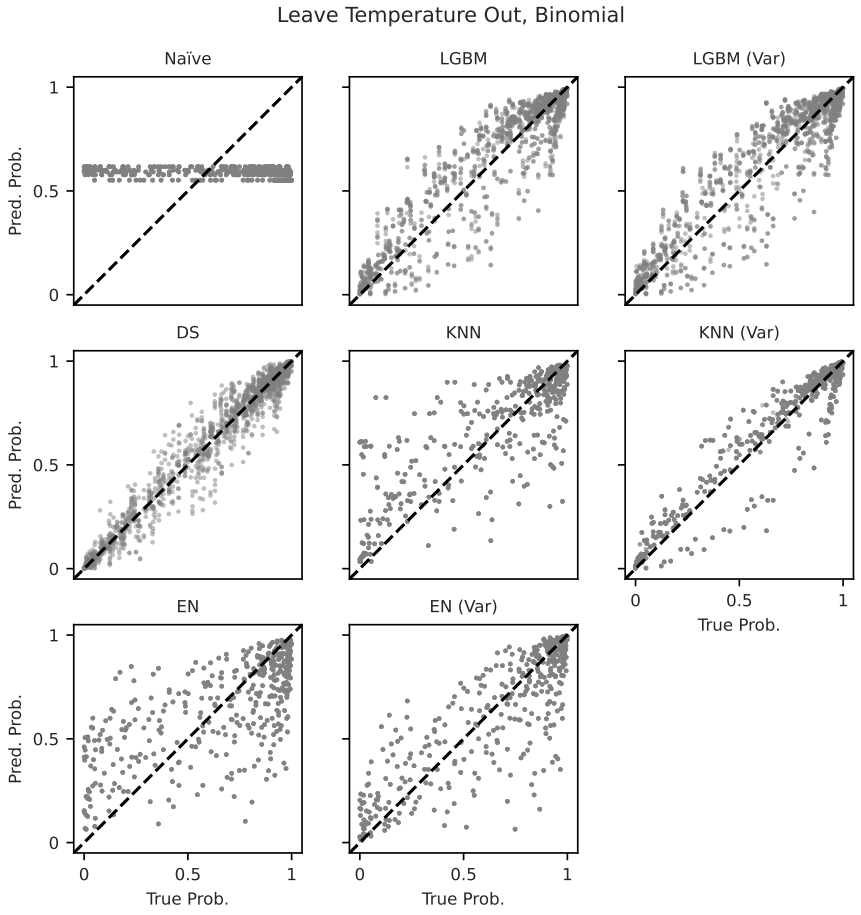

Figure E5: True *vs.* predicted probabilities for each model using nested **leave-one-temperature-out** cross-validation and the Binomial loss.

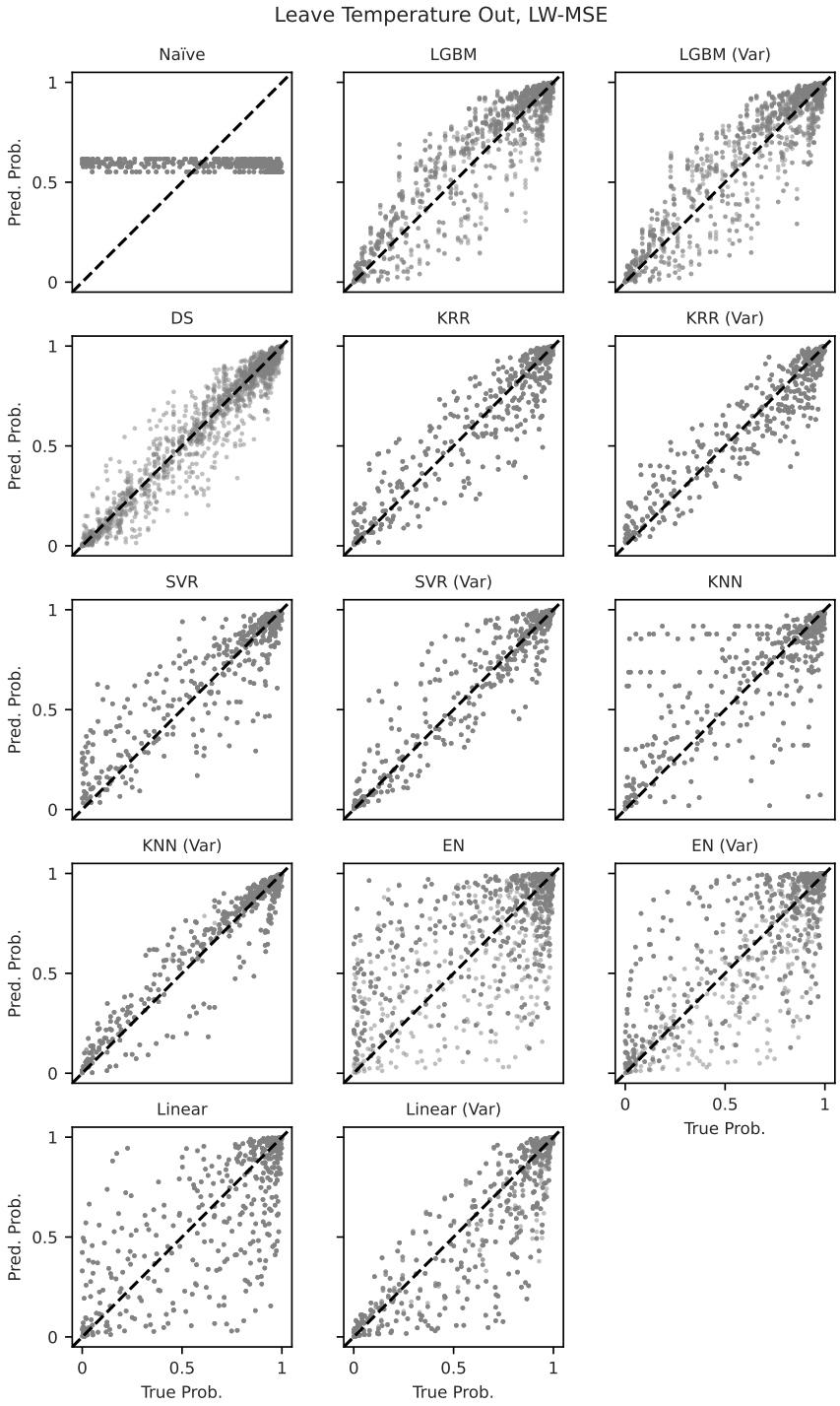

Figure E6: True *vs.* predicted probabilities for each model using nested **leave-one-temperature-out** cross-validation and the LW-MSE loss.

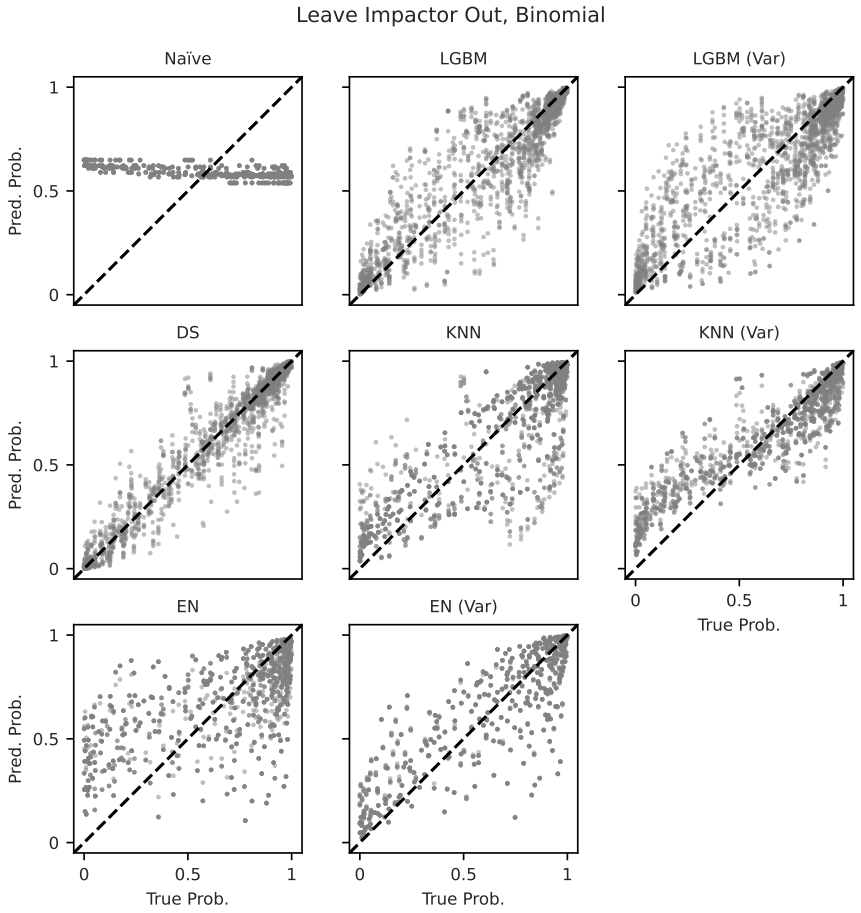

Figure E7: True *vs.* predicted probabilities for each model using nested **leave-one-impactor-out** cross-validation and the Binomial loss.

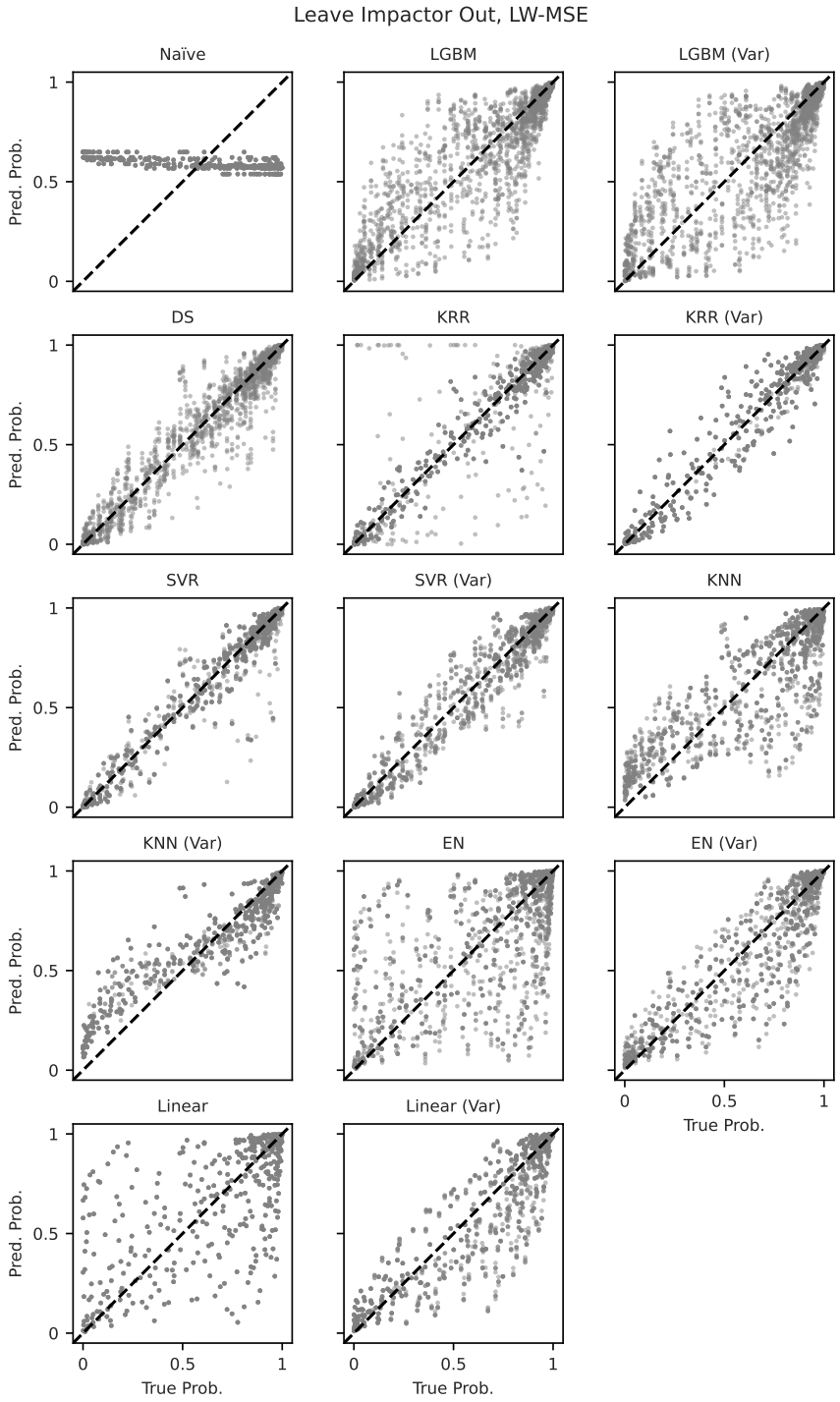

Figure E8: True *vs.* predicted probabilities for each model using nested **leave-one-impactor-out** cross-validation and the LW-MSE loss.

F Example Input and Output

Cluster Descriptors				**Impactor Descriptors**				**Environmental Descriptors**	**Outputs**
#Si	#h	#e_1	#e_2	#Si	#h	#e_1	#e_2	Temperature (K)	Probability
4	0	8	0	1	1	3	0	300	0.979
2	6	0	0	1	4	0	0	600	0.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table F1: Example inputs and outputs for the machine learning models (assuming no pre-processing). # Si and # h indicate the number of silicon and hydrogen atoms, respectively. # e_1 and # e_2 indicate the first and second elements in the vector #**e** describing the number of unpaired electrons per silicon atom.