Probing Accuracy-Speedup Tradeoff in Machine Learning Surrogates for Molecular Dynamics Simulations

Fanbo Sun, JCS Kadupitiya, and Vikram Jadhao*

Intelligent Systems Engineering, 700 N. Woodlawn Avenue, Indiana University, Bloomington, Indiana 47408

E-mail: vjadhao@iu.edu

Abstract

The performance promise of machine learning surrogates of molecular dynamics simulations of soft materials is significant, but generally comes at the cost of acquiring large training datasets to learn the complex relationships between input soft material attributes and output properties. Under the constraint of limited high-performance computing resources, optimizing the size of the training datasets becomes paramount. Using an artificial neural network based surrogate for molecular dynamics simulations of confined electrolytes, we explore the tradeoff between surrogate accuracy and computational gains. Accuracy is assessed by computing the root mean square errors between the surrogate predictions and the ground truth results obtained via molecular dynamics simulations. The computational performance is judged by evaluating the speedup which incorporates the training dataset creation time. Improvement in accuracy occurs with a loss of speedup, which scales as the inverse of the training dataset size. The link between surrogate generalizability and the accuracy-speedup tradeoff is assessed by examining the errors incurred in surrogate predictions on unseen, interpolated input variables and developing a net speedup metric to capture the associated gains.

1 Introduction

Molecular dynamics (MD) simulations are powerful computational methods for investigating the microscopic origins of a wide variety of material and chemical phenomena. These simulations furnish molecular-level mechanisms that drive structure and property control in materials while isolating interesting regions of the material design space to aid experimental exploration and discovery. Recent years have seen a surge in the integration of machine learning (ML) methods with MD simulations to reduce their computational costs, enhance their predictive power, and expedite the analysis of high-dimensional output data.[1–14] A number of studies have explored the use of ML to develop surrogates for MD simulations.[9–12,15,16] The key idea behind a surrogate is to collect data from conventional MD simulations and train an ML model that approximates the relationships between the input parameters and the simulation outcomes.[9–12] Thus, the surrogate bypasses part or all of the explicit evolution of the simulated components. The associated performance enhancement enables the surrogate to serve as a fast exploratory tool that complements the MD simulation in traversing the input design space, and to act as a dynamic alternative to simulation caching for retrieval of reliable estimates of simulation outputs.

Neural networks, including deep neural networks (DNNs), have proven to be particularly effective in the design of surrogates. Examples include DNNs that predict adsorption equilibria for different thermodynamic conditions,[11] DNN-based denoising autoencoders that predict the temporally-averaged radial distribution function of Lennard-Jones fluids from a single snapshot of fluid particles generated in MD simulations,[9] Bayesian neural networks that predict the dissociation timescale of compounds bypassing the explicit time evolution of the particle trajectories in *ab initio* MD simulations,[10] and autoencoders that generate new protein-like structures and act as a proxy for MD simulations to mine the protein conformational space.[17]

In our previous work,[15,16,18] we introduced ML surrogates for MD simulations of soft materials. Our goal was to demonstrate that artificial neural network (ANN) based regression models can accurately predict the relationships between the input parameters characterizing the soft-matter system and the simulation outcomes describing the system's equilibrium properties. The approach was illustrated with the design of an ANN-based surrogate for coarse-grained MD simulations of confined electrolytes. These simulations establish the links between the distributions of electrolyte ions and the ion attributes for diverse solution conditions. The ionic distributions shed light on the origins of ion-specific effects in confinement (e.g., ion adsorption at interfaces), which can be meaningful in the interpretation of effective interactions between nanoparticles, biomolecules, and membranes, and for the evaluation of interfacial activity in separation technologies.[19] These distributions also provide a reliable guide to the regions of the material design space where significant changes in the structural organization of ions are expected, which can aid the experimental exploration and design of electrolyte-based materials.[20–25] The ML surrogate was trained to predict the relationship between the output distribution of positive ions and the input variables characterizing the electrolyte solution comprising positive and negative ions of the same size, confined by uncharged surfaces. The predicted ionic density profiles were in excellent agreement with MD simulations.[15,16] Additionally, the time required for predictions using the surrogate was significantly smaller (by a factor of 10,000) compared to the runtime of the corresponding MD simulation.

In general, once a surrogate is trained, we can obtain outputs through ML inference in seconds, instead of running an MD simulation, which usually takes much longer (e.g., hours). The high accuracy and small inference time allude to the significant scientific and computational performance promise of surrogates for MD simulations of soft materials. However, the performance enhancement comes at the cost of generating large training datasets, which is a time-consuming process that requires running many MD simulations on high-performance computing (HPC) clusters. Designing ML surrogates that reach an acceptable level of scientific performance (accuracy) with large gains in the computational performance requires an understanding of the link between the accuracy-speedup tradeoff and the size of the training datasets. Sample size determination for training and testing sets has long been recognized as a critical task in traditional ML applications such as the design of high-performance pattern recognition systems,[26] which consider the tradeoff between the degree of precision and limitations on resources.[27] Recently, the importance of designing optimal training and validation datasets for developing robust ML models has been recognized in the biomedical engineering and materials science domains.[28–30]

In this paper, we study the accuracy-speedup tradeoff associated with surrogate models using the ML surrogates for MD simulations of confined electrolytes. Surrogates are designed using a dataset generated by conducting simulations of 4050 different electrolyte systems that exhibit a much greater complexity in the relationship between the input electrolyte attributes and the output ionic structure due to the inclusion of ionic size asymmetry, charged surfaces, and a broader range of concentrations than previously explored. Surrogates are tasked to predict the density profiles of both positive and negative ions by learning the relationship between 1004 output features characterizing the density profiles and 5 input features: confinement length, electrolyte concentration, positive ion diameter, negative ion diameter, and surface charge density. The scientific performance is measured by computing the root mean square errors (RMSE) between the surrogate predictions and the ground truth results obtained via MD simulations, as well as by comparing the output features obtained via the two approaches. The computational performance is judged by evaluating the speedup which incorporates the training dataset creation time. Two data reduction methods: random splitting and deterministic separation, are utilized to study the surrogate performance and generalizability. A study of the changes in the surrogate accuracy with the training dataset size $N_{\text{train}} \in (150, 3550)$ reveals a power-law decrease in the overall RMSE, and the onset of convergence of the surrogate accuracy for $N_{\text{train}} \gtrsim 1550$ samples. The speedup decreases with increasing N_{train}, scaling as $1/N_{\text{train}}$. Based on the overall and output-specific errors and the agreement between the predicted density profiles and the ground truth results obtained via MD simulations, an acceptable level of accuracy under the constraint of maximizing the speedup is reached for the training dataset containing $N_{\text{train}} = 1550$ samples. Surrogate generalizability is assessed by examining the surrogate performance on input variable values obtained via interpolation. The surrogate performance varies greatly depending on which input material attribute is hidden, and the fraction of the samples associated with the interpolated values the surrogate sees during training. Increasing this fraction improves the accuracy but at the cost of reducing the potential of computational gains. A simple metric for the net speedup is presented to probe this accuracy-speedup tradeoff.

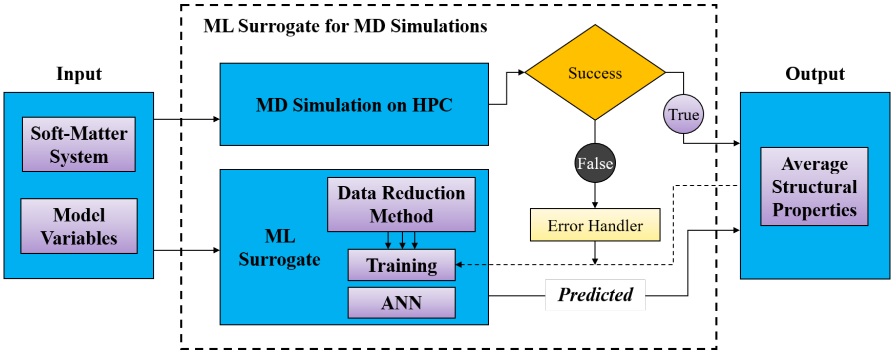

Figure 1: Overview of the approach to design, train, and use an ML surrogate for MD simulation of a soft-matter system.

2 Methods

Figure 1 shows the overview of the approach describing the design, training, and use of a surrogate for MD simulation of a soft-matter system. We first run MD simulations on an HPC cluster for different model variables (input) characterizing the soft-matter system, and save the converged simulation outcomes (output) for training the ML surrogate, which occurs after a set number of successful simulation runs. Error handler aborts the MD simulation program and displays appropriate error messages when a simulation fails due to any pre-defined criteria. The inputs are also fed to the ML-based prediction module, which is trained to learn the associated output quantities. Once trained, the ML surrogate is ready to be used for predicting the output properties of a soft-matter system characterized with a given set of input material attributes. Figure 1 highlights the use of data reduction methods to train the surrogates.

2.1 Input Variables and Output Quantities

We consider a monovalent electrolyte in water confined by two planar interfaces at room temperature $T = 298$ K. A coarse-grained model[19,31–34] is employed to describe the electrolyte solution. Water is modeled as an implicit solvent with a relative dielectric permittivity of 80. The main simulation cell is a rectangular box with dimensions $L \times L \times h$, where L denotes the box edge length in the unconfined x and y directions, and h denotes the confinement length (interfacial separation). The box dimension L is selected based on the Debye length of the solution;[19] for electrolyte concentration $c > 0.5$ M, $L = 10$ nm, and for $c \leq 0.5$ M, $L = 15$ nm. h ranges from $4 - 5$ nm. The planes at $z = -h/2$ and $z = h/2$ represent the location of the charged interfaces ($z = 0$ corresponds to the midpoint between the interfaces). Each interface is characterized with a uniform charge density $\sigma_s < 0$, which is modeled by discretizing the interface with M mesh points and assigning each mesh point the same charge $q = \sigma_s L^2/M$. $M = 784$ for $c > 0.5$ M, and $M = 1764$ for $c \leq 0.5$ M. The positively-charged ions (cations) and negatively-charged ions (anions) associated with the monovalent electrolyte are modeled as spheres with hydrated sizes d_+ and d_- respectively. An appropriate number of counterions, modeled as cations of the same diameter and charge, are included in the confinement to ensure electroneutrality. The total number of ions within the confinement ranges from 366 to 1228, depending on the concentration, confinement length, and the surface charge density characterizing the interfaces.

Electrolyte system attributes h, c, d_+, d_-, and σ_s are chosen as the surrogate input variables (Table 1). c is defined as $c = N_-/V$, where N_- is the number of anions and V is the volume of the simulation box. We note that not all electrolyte system attributes that are expected to alter the output ionic structure are considered as tunable input variables. For example, ion valencies (set to $1, -1$), temperature (298 K) and solvent permittivity (80) are held fixed across all the simulations.

MD simulations are performed using LAMMPS[35] in an NVT ensemble at $T = 298$ K. Ion-ion and ion-interface steric interactions are modeled using Lennard-Jones potentials,[19] and all electrostatic interactions are modeled using Coulomb potentials whose long-range is properly treated with Ewald sums.[36] Each system is simulated for 1 ns to reach equilibrium with a timestep of 1 femtosecond. After equilibration, systems are simulated for $\approx$ 9 ns, and ion trajectory data are collected every 0.1 ps. Due to the planar symmetry and the homogeneously-charged interfaces, the ionic distributions vary only in the direction perpendicular to the interfaces, and are functions of a single variable z. Trajectory data samples are used to compute the average ionic distributions $n_+(z)$ of cations and $n_-(z)$ of anions, which form the output quantities. Converged results for the ionic distributions are obtained by computing the average using the samples generated post equili-

Table 1: Input variables and their values at which simulations are launched to generate the dataset

Input Variable	Simulated Values
Confinement length h (nm)	4.0, 4.2, 4.4, 4.6, 4.8, 5.0
Electrolyte concentration c (M)	0.1, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0
Cation diameter d_+ (nm)	0.2, 0.3075, 0.415, 0.5225, 0.63
Anion diameter d_- (nm)	0.2, 0.3075, 0.415, 0.5225, 0.63
Surface charge density σ_s $(\mathrm{C/m^2})$	−0.01, −0.015, −0.02

bration from 3 ns to 10 ns. Each distribution or number density profile is specified by computing the average ion population densities (with error bars) at 502 locations within the confinement region $z \in (-h/2, h/2)$. The output of the ML surrogate is thus characterized with 1004 features.

2.2 Datasets for Surrogate Training

Table 1 shows the region of the material design space that contributes toward the dataset used to train and test the ML surrogate. By sweeping over the shown discrete values of each input variable h, c, d_+, d_-, and σ_s, 4050 unique electrolyte systems are created. For each of these systems, MD simulations are run and the converged distributions of cations and anions are extracted as output. Each MD simulation is performed for ≈ 10 nanoseconds and takes ≈ 3.5 hours using 96 cores. Generating the entire dataset took approximately 20 days, including the queue wait times on the Indiana University BigRed3 supercomputing cluster.

Figure 2 illustrates the two data reduction methods employed in our investigation to prepare the training and testing datasets: random splitting and deterministic separation. Figure 2(a) illustrates the random splitting method used in results shown in Sections 3.1 and 3.2, where the dependence of the surrogate performance on the size of the training dataset is investigated. First, $N_{\text{test}} = 500$ samples are randomly drawn from the total dataset S of size $N_{\text{total}} = 4050$ to form an independent test dataset S_{test}. The samples in this test dataset are hidden from the surrogate. Next, N_{train} samples are randomly drawn from the reduced dataset S_R of size $N_{\text{total}} - N_{\text{test}} = 3550$ to form the train dataset S_{train}. This process is used to create 15 training datasets with sample size $N_{\text{train}} = 150, 200, 250, 300, 350, 400, 450, 500, 550, 1050, 1550,\ 2050, 2550, 3050, 3550$. Post-training, the

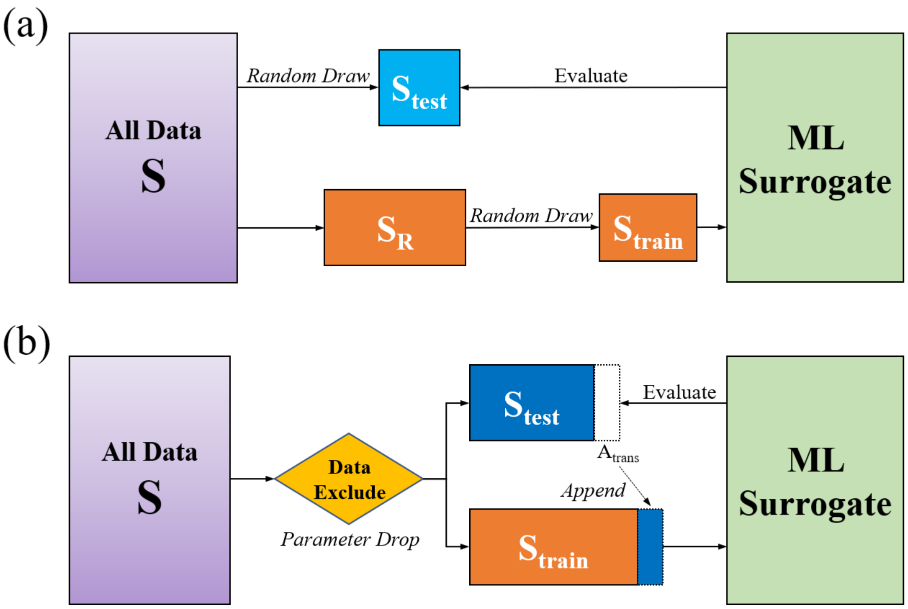

Figure 2: Overview of the data reduction methods used to train the surrogate. In the random splitting method (a), a test dataset S_{test} is first formed by randomly drawing samples from the total dataset S. The train dataset S_{train} is formed by randomly drawing samples from the reduced dataset S_R. In the deterministic separation method (b), a decision is first made to exclude the samples associated with pre-selected input parameters to create the test dataset S_{test}. The remaining data forms the train dataset S_{train}. If needed, A_{trans} samples are removed from S_{test} and appended to S_{train}.

surrogate is evaluated using the N_{test} samples in the test dataset S_{test}.

Figure 2(b) shows the deterministic separation method used in results shown in Section 3.3, where the generalization ability of the surrogate is assessed. First, a decision is made about what input variable and associated values are hidden from the training dataset. An example of such a decision is to exclude all electrolyte systems having the concentration value $c = 1.0$ M from the training of the surrogate. These systems form the test dataset S_{test} of size N_{test} such that each sample $s \in S_{\text{test}}$ is an input-output pair associated with a simulation of an electrolyte at $c = 1.0$ M. The training dataset S_{train} is the complement of S_{test}, i.e., $S_{\text{train}} = \{s \in S : s \notin S_{\text{test}}\}$. In the aforementioned example, the surrogate gets trained on input-output pairs associated with all electrolyte systems except those that are characterized with $c = 1.0$ M. Such an approach enables the evaluation of the surrogate predictions for input variable values not “seen” during training. In order to study the link between the surrogate performance and the number of samples the surrogate sees during training that have the hidden input value, we randomly draw and remove

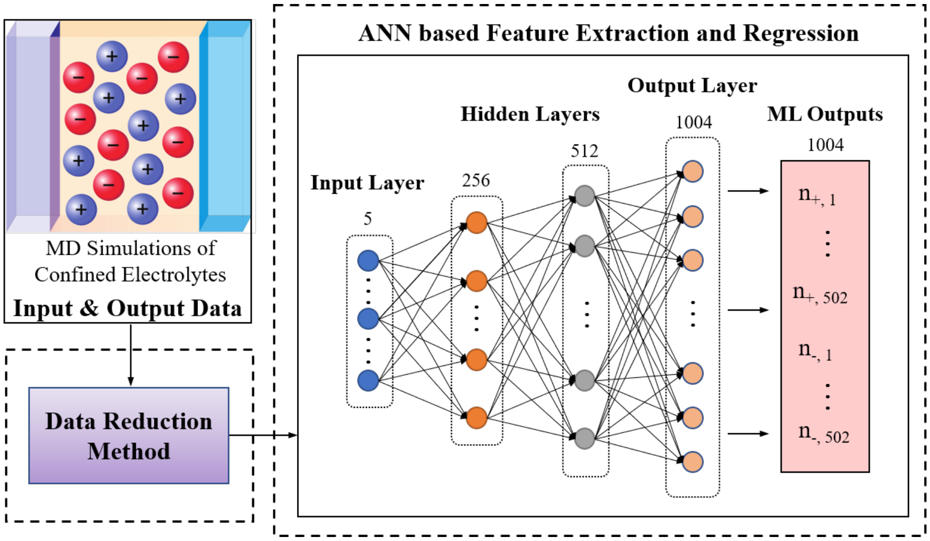

Figure 3: Artificial neural network (ANN) based regression model used in the ML surrogate to extract features and predict output density profiles of cations (n_+) and anions (n_-). The input layer, two hidden layers, and output layer are characterized with 5, 256, 512 and 1004 nodes respectively.

A_{trans} samples from the test dataset S_{test} and append it to the training dataset S_{train}. This enables us to systematically investigate how the surrogate's performance is affected by changing A_{trans}, with $A_{\text{trans}} = 0$ corresponding to the case where the surrogate makes predictions in a complete "blind" mode (see Section 3.3 for more details).

2.3 Feature Extraction and Regression

The ML surrogate is trained to predict the density profiles of cations and anions. Each density profile is characterized with 502 points. For a given sequence of input parameters, the surrogate thus makes a total of $P = 1004$ predictions to quantify the output ionic distributions. Figure 3 shows a sketch of the ANN model used in the surrogate to implement the regression and prediction of these output quantities. The ANN architecture has 2 hidden layers, similar to the surrogate employed in our earlier work.[16] The weights and biases in these hidden layers are determined by the regression process, following an error backpropagation algorithm, implemented by a stochastic gradient descent procedure. This process uses a training dataset and an appropriate loss function for error computation and backpropagation to update the weights and biases after each batch of input data is regressed through the network by a simple forward prediction. The mean square error (MSE) between the ground truth and the surrogate predictions is chosen as the loss function.

Training the ANN model involves an appropriate selection of hyperparameters such as the number of first hidden layer units n_1, the number of second hidden layer units n_2, learning rate l_r, batch size b, and the number of epochs n_e. l_r acts as a step size associated with the gradient descent process to reach the minimum of the MSE loss function. b is the number of training samples allowed to pass through the ANN before updating its weights and biases. n_e controls the number of complete passes made through the entire training dataset.

In both random splitting and deterministic separation methods, the data in the training dataset S_{train} is separated further into training and validation sets using a ratio of $0.8 : 0.2$ in order to find the optimal hyperparameters for the ANN model. A min-max normalization filter is applied to normalize the input data at the preprocessing stage. A separate grid search is performed for each training dataset S_{train} to obtain the set of optimal hyperparameters by examining the validation loss. The grid search is carried out for a total of $n_e = 20000$ epochs for the following hyperparameters: $n_1 \in \{128, 256, 512\}$, $n_2 = \{256, 512\}$, $l_r \in \{0.0001, 0.0002\}$, $b \in \{32, 64\}$. Regardless of the training dataset size, the optimal values are found to be $n_1 = 256$, $n_2 = 512$, $b = 32$. For most cases, the optimal value for the learning rate is $l_r = 0.0001$. A few training datasets do yield a marginally lower value of the validation loss for $l_r = 0.0002$, however, the ANN performance on the test dataset is unaffected when $l_r = 0.0001$ value is used.

The ReLU activation function is applied to the output of the input and the second hidden layers, while the sigmoid activation function is applied to the output of the first hidden layer. The Adam optimizer is used to optimize the error backpropagation process. During the forward propagation in the training phase, the dropout rate in the dropout layers between the input and the first hidden layer, and between the first and second hidden layers is set to $d_r = 0.1$ to prevent overfitting. The weights in each hidden layer are initialized using a Glorot normal distribution characterized with a mean of 0 and a variance of $2/(h + h')$, which changes according to the size of the input (h) and output (h') associated with the hidden layer. The ANN model is implemented using scikit-learn and TensorFlow Keras libraries. Scikit-learn is used for grid search and feature scaling, and Tensorflow Keras is used to build, train and evaluate the ANN model.

3 Results and Discussion

3.1 Surrogate Performance vs Train Dataset Size

We begin by showing the results for the surrogate model convergence for different training dataset sizes generated via the random splitting method depicted in Figure 2(a). Recall that in this method, an independent test dataset S_{test} of size $N_{\text{test}} = 500$ is first created by randomly drawing elements from the total dataset S of size $N = 4050$. The reduced dataset S_R of size $N_{\text{total}} - N_{\text{test}} = 3550$ is used to create training datasets S_{train} of different sizes $N_{\text{train}} = \{150, 200, 250, \ldots, 3050, 3550\}$. Each element in S_{train} is randomly drawn from S_R. For each S_{train}, the ANN model convergence is examined by computing the validation loss L for each epoch of training and examining the overfitting behavior. L is computed as the average mean square error (MSE) incurred by the model to make $P = 1004$ predictions describing the cation and anion density profiles associated with the validation dataset $S_{\text{val}} \in S_{\text{train}}$ of size $N_{\text{val}} = 0.2N_{\text{train}}$:

$$L = \frac{1}{N_{\text{val}}P} \sum_{j=1}^{N_{\text{val}}} \sum_{k=1}^{P} \left| \hat{y}_j^k - y_j^k \right|^2 . \tag{1}$$

Here, $\hat{y}_j^k$ represents the k^{th} prediction made by the surrogate to characterize the ion number density for the electrolyte system j, and y_j^k is the corresponding ground truth result.

The validation loss L decreases with increasing training dataset size N_{train} from 150 to 3550. For the sake of clarity, Figure 4 shows a comparison of L for 5 datasets of sizes $N_{\text{train}} = 150, 350, 500, 1050, 3550$. For $N_{\text{train}} > 300$, L exhibits a decrease with increasing number of epochs up to the highest value of $n_e = 20000$, yielding convergence for $n_e > 15000$. For $N_{\text{train}} < 300$, a small increase in L is observed within $15000 < n_e < 20000$, indicating the crossover to the overfitting regime. For example, for the dataset of size $N_{\text{train}} = 150$, the crossover occurs near

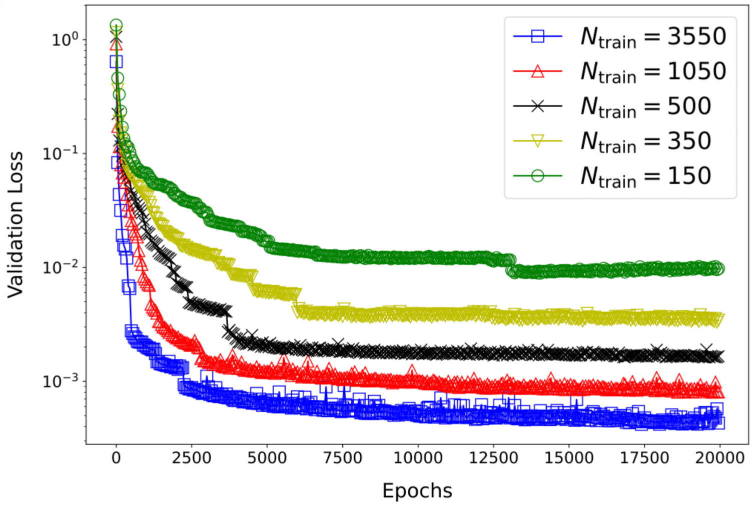

Figure 4: Validation loss defined in Equation 1 vs number of epochs n_e for training dataset size $N_{\text{train}} = 150$ (green circles), 350 (yellow down triangles), 500 (black cross), 1050 (red up triangles), 3550 (blue squares). For $N_{\text{train}} = 350, 500, 1050, 3550$, the validation loss decreases with increasing n_e and exhibits convergence for $n_e > 15000$. For $N_{\text{train}} = 150$, an increase in the validation loss is observed when $n_e > 15000$, signaling overfitting.

$n_e = 15000$, which corresponds to a validation loss of $L = 9.31 \times 10^{-3}$. The crossover value for n_e increases slightly with increasing $N_{\text{train}} = 200, 250, 300$. For simplicity, we checkpointed all models at $n_e = 15000$ and saved the associated weights and biases to evaluate the performance of the surrogate on the independent test dataset. On average, changing the number of epochs $n_e \in (15000, 20000)$ had an insignificant effect on the surrogate performance.

The scientific performance (accuracy) of the surrogate is assessed by examining the errors incurred in the surrogate predictions for the cation and anion density profiles associated with the electrolyte systems in the unseen test data. The root mean square error (RMSE) E_k associated with the k^{th} prediction characterizing the density profiles is computed by averaging over the errors incurred in making this prediction for all the N_{test} samples in the test dataset:

$$E_k = \left(\frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{test}}} \left| \hat{y}_j^k - y_j^k \right|^2 \right)^{1/2}. \tag{2}$$

Here, $\hat{y}_j^k$ is the k^{th} prediction or inference associated with the ion number density for the input system specified by the index j, and y_j^k is the corresponding ground truth. Prediction numbers

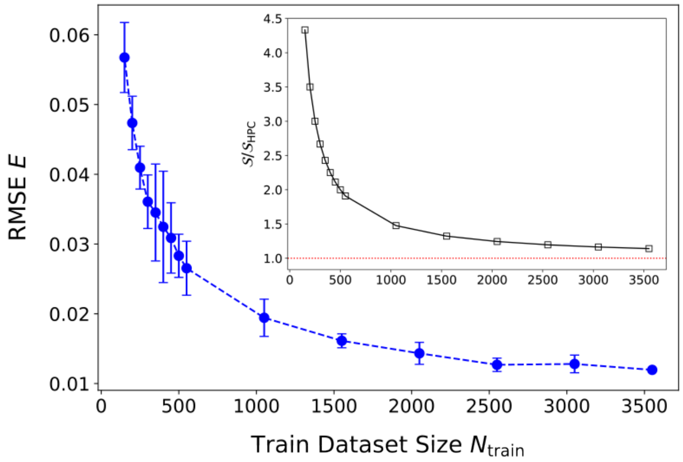

Figure 5: Overall RMSE value E (in units of M) defined in Equation 3 decreases as a power-law with increasing the size N_{train} of the surrogate training dataset. A sharp decrease is observed when N_{train} increases from 150 to 1050. E starts to exhibit convergence for $N_{\text{train}} \gtrsim 1550$. The inset shows the speedup $\mathcal{S}/\mathcal{S}_{\text{HPC}}$ defined in Equation 5 vs N_{train}. The dotted red line represents $\mathcal{S} = \mathcal{S}_{\text{HPC}}$. $\mathcal{S}/\mathcal{S}_{\text{HPC}}$ decreases with increasing N_{train}, scaling as $1/N_{\text{train}}$.

$k = 1, 2, ..., 502$ correspond to the cation density profile and $k = 503, 504, ..., 1004$ correspond to the anion density profile. For a train dataset S_{train} of size N_{train}, E_k is evaluated using Equation 2 for each of the $P = 1004$ surrogate predictions and an overall RMSE E is computed:

$$E = \frac{1}{P}\sum_{k=1}^{P} E_k. \tag{3}$$

E serves as a metric to evaluate the accuracy of the surrogate as a function of the train dataset size. In what follows, all reported RMSE values (e.g., E_k, E) are in units of M.

Figure 5 shows the overall RMSE E for the 15 training datasets described in Section 2.2. E exhibits a power-law decrease with increasing training dataset size N_{train}. A steep drop in E from $N_{\text{train}} = 150$ to 1050 is followed by a relatively milder decay as N_{train} is increased further, and the surrogate accuracy reaches convergence for $N_{\text{train}} \gtrsim 1550$ samples. In order to evaluate the robustness of the surrogate predictions, we computed the error bar associated with E by employing the random splitting method 10 times to get 10 different S_{train} datasets for the same N_{train}. In general, the error bar is larger for $N_{\text{train}} < 500$, indicating a low degree of robustness for the predictions made by the surrogate trained on smaller number of samples.

The computational performance of the surrogate is assessed by evaluating the potential gain or speedup resulting from the surrogate use. The speedup depends on the computational costs associated with the creation of the training dataset S_{train}, and can be expressed as:

$$\mathcal{S} = \frac{N_p t_{\text{seq}}}{N_p t_p + N_{\text{train}} t_{\text{train}}} \tag{4}$$

where N_p is the number of surrogate predictions, N_{train} is the number of elements in the training dataset, t_{seq} is the time to run the MD simulation via the sequential (serial) model, t_p is the time it takes for the surrogate to make a prediction for one input, and t_{train} is the average walltime associated with the MD simulation to create one element of S_{train}. t_{train} is typically similar to the average runtime of the parallelized MD simulation. $N_{\text{train}} \times t_{\text{train}}$ is the amount of time utilized to create the training dataset. For MD simulations of confined electrolytes, $t_{\text{seq}} \approx 24$ hours, $t_{\text{train}} \approx 3.5$ hours, and $t_p \approx 0.3$ seconds. Note that in Equation 4, we have assumed that the training dataset generation using MD simulations is a much more time-consuming process compared to the surrogate training using TensorFlow, which is generally the case.

The above formula highlights a unique feature of the ML surrogate performance: $\mathcal{S}$ rises with increasing N_p, that is, the speedup increases as the surrogate makes more predictions. At first glance, this suggests a limitless computational performance gain through the use of the surrogate. However, the predictions are only useful if they meet an acceptable level of scientific performance or accuracy. Considering this accuracy-speedup tradeoff, a more practical assessment of the speedup $\mathcal{S}$ is the gain achieved in making predictions with errors similar or less than the overall RMSE value E, where the latter serves as an acceptable accuracy level, which changes with train dataset size N_{train}. In general, the errors incurred in the surrogate predictions on the train dataset are expected to be smaller than the RMSE value E. Thus, an estimate for $\mathcal{S}$ is obtained by setting

$N_p = N_{\text{test}} + N_{\text{train}}$ in Equation 4:

$$\mathcal{S} = \frac{N_p t_{\text{seq}}}{N_p t_p + N_{\text{train}} t_{\text{train}}} \approx \mathcal{S}_{\text{HPC}} \frac{N_p}{N_{\text{train}}} = \mathcal{S}_{\text{HPC}} \left(1 + \frac{N_{\text{test}}}{N_{\text{train}}} \right), \tag{5}$$

where the second equality follows by noting $t_p \approx 0 \ll t_{tr}$ and we have introduced $\mathcal{S}_{\text{HPC}} = t_{\text{seq}}/t_{\text{train}}$ to denote the traditional speedup obtained by parallelizing the MD simulation using HPC resources. For the case of simulations of confined electrolytes considered here, $\mathcal{S}_{\text{HPC}} = 7$.

Figure 5 inset shows the speedup $\mathcal{S}/\mathcal{S}_{\text{HPC}}$ associated with the same 15 training dataset sizes N_{train} for which the error E is shown in the outset. $\mathcal{S}$ scales as $1/N_{\text{train}}$ and decreases from $\approx 4.3\,\mathcal{S}_{\text{HPC}}$ for $N_{\text{train}} = 150$ to $\approx 1.14\,\mathcal{S}_{\text{HPC}}$ for $N_{\text{train}} = 3550$. This trend highlights the tradeoff between the surrogate accuracy and the potential for speedup resulting from its application. A gain in the accuracy with increasing train dataset size occurs at a loss in the speedup. Note that $\mathcal{S}/\mathcal{S}_{\text{HPC}} > 1$, as evident by all $\mathcal{S}$ values above the dotted red line indicating $\mathcal{S} = \mathcal{S}_{\text{HPC}}$. This indicates that for all training dataset sizes, the speedup from the use of the surrogate exceeds the enhancement resulting from parallelization.

It is likely that the number of predictions made by a well-trained surrogate with an acceptable level of average error E will exceed $N_{\text{test}} + N_{\text{train}}$ samples. For example, the speedup can be boosted by tasking the surrogate to make predictions on the interpolated values between the discretized input variables. This requires an assessment of the associated errors incurred by the surrogate that are linked to its generalizability, which we discuss in Section 3.3.

We now compare the cation and anion number density profiles predicted by the ML surrogate with the ground truth results obtained using MD simulations for the unseen electrolyte systems in the test dataset in order to obtain a direct assessment of the prediction quality. Figure 6 shows the results of cation (outset) and anion (inset) density predictions for 4 representative input systems randomly selected from the test dataset. The systems are labeled $(h, c, d_+, d_-, \sigma_s)$ using the 5 input variables defined in Section 2.1. The 4 systems are: system I $(4.6, 0.25, 0.5225, 0.415, -0.01)$, system II $(4.4, 1.0, 0.3075, 0.63, -0.01)$, system III $(4.6, 1.75, 0.3075, 0.63, -0.01)$, and system IV

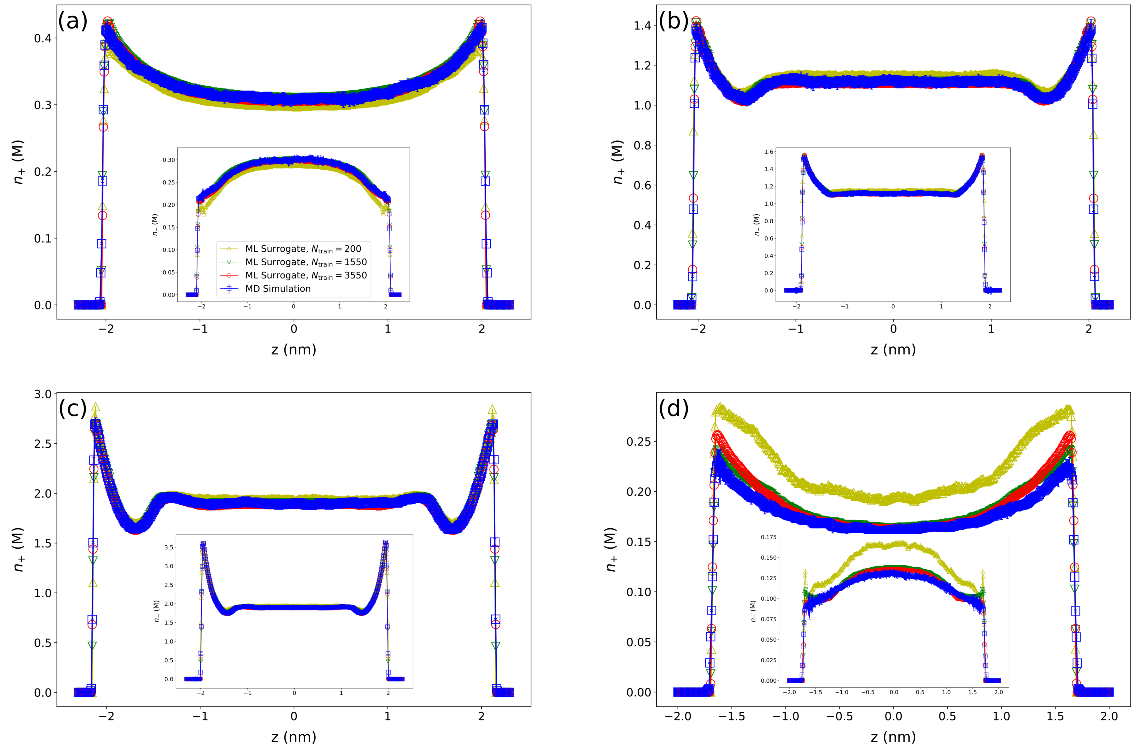

Figure 6: Cation and anion (inset) density profiles for four representative electrolyte systems I (a), II (b), III (c), and IV (d) predicted by the surrogate. Ground truth results (blue squares) are extracted using MD simulations. Surrogate predictions are shown for train dataset size $N_{\text{train}} = 200$ (yellow up triangles), 1550 (green down triangles), and 3550 (red circles). For systems I-III, the surrogate trained with $N_{\text{train}} = 1550$ and 3550 samples produces results in good agreement with the ground truth, while surrogate predictions for $N_{\text{train}} = 200$ are inferior. For system IV, all surrogate predictions deviate from the ground truth. See main text for the electrolyte system details.

$(4, 0.1, 0.63, 0.5225, -0.01)$. Figure 6 (a), (b), (c), and (d), respectively, show surrogate prediction results for systems I, II, III, and IV for three train dataset sizes $N_{\text{train}} = 200, 1550, 3550$.

For systems I, II and III, surrogates designed using $N_{\text{train}} = 1550, 3550$ samples produce density profiles in good agreement with the ground truth, while the surrogate trained with $N_{\text{train}} = 200$ samples generates inferior predictions. For system IV, all three surrogates yield predictions that deviate away from the ground truth. The surrogate trained using $N_{\text{train}} = 200$ samples fails completely to capture the ionic structure. An explanation emerges by examining the input variables for the electrolyte system IV, many of which are on the edge of the design space used to train the surrogate (for example, $h = 4.0$ nm, $c = 0.1$ M, $d_+ = 0.63$ nm). As N_{train} decreases, the surrogate performance worsens because it does not “see” enough of these edge combinations during training.

3.2 Output-Specific Surrogate Performance

The negatively-charged surfaces, that tend to attract cations and repel anions, and the differences in the size of cations and anions lead to differences in the cation and anion density profiles, as evident in Figure 6. In addition to the overall RMSE E, it is thus useful to examine the cation-specific and anion-specific accuracy values for a more precise evaluation of the surrogate performance. The density profiles also show that the confinement created by the two interfaces produces distinct ion accumulation and depletion behaviors within the interfacial regions as compared to the bulk. Therefore it is also useful to assess the performance of the surrogate for different regions within the confinement that exhibit distinct ionic structure. In this sub-section, we carry out a detailed examination of these output-specific surrogate performance metrics.

We introduce the set $K^+ = \{1, 2, 3...502\}$ comprising prediction indices associated with the cation density profile and define $E_k^+ = E_k$ for $k \in K^+$ as the average RMSE value E_k incurred in the k^{th} prediction characterizing the cation density profile. Similarly, we introduce $K^- = \{503, 504...1004\}$ which comprises prediction indices associated with the anion density profile, and define $E_k^- = E_k$ for $k \in K^-$ as the average RMSE value incurred in predictions characterizing the anion density profiles. For both K^+ and K^- sets, it is also useful to note that the low and high index values represent the confining interfaces, and the indices outside these ranges are defined as associated with the bulk of the confined region.

Figure 7 shows the plot of E_k^+ and E_k^- associated with predictions made by the surrogate trained with $N_{\text{train}} = 200, 1550, 3550$ samples. To facilitate the comparison of the errors for cation and anion density predictions, we left-shift the prediction index numbers for anions by 502, i.e., the prediction indices $k = \{503, 504, \ldots, 1004\}$ for the anion density are mapped to $k = \{1, 2, \ldots, 502\}$. In general, RMSE values for both cation and anion density predictions are higher near the interface compared to the bulk. Very large errors are observed for the case of $N_{\text{train}} = 200$ near the interface. Both E_k^+ and E_k^- decrease as N_{train} is increased.

To understand the performance of the surrogate on different regions of the confinement that are associated with distinct ionic structure, we evaluate the contributions to the RMSE emerging

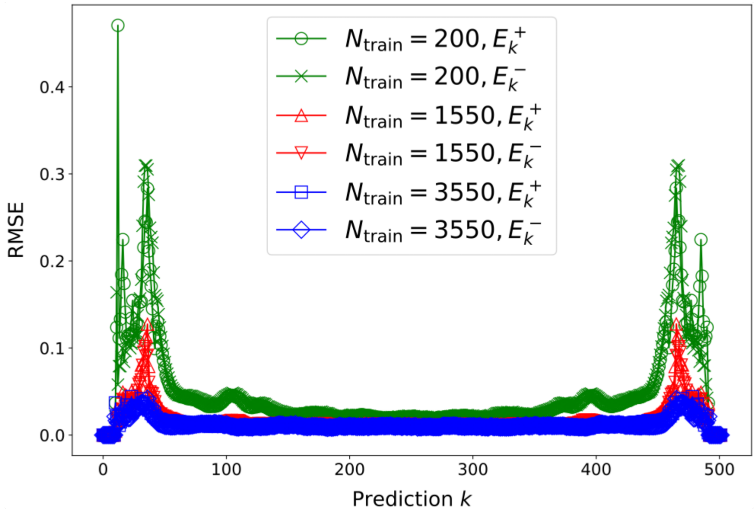

Figure 7: RMSE values (in units of M) associated with surrogate predictions for the cation density profile (E_k^+) and anion density profile (E_k^-) vs prediction index k. Green circles and crosses, red up and down triangles, blue squares and diamonds represent E_k^+ and E_k^- for training dataset size $N_{\text{train}} = 200, 1550, 3550$ respectively. Large errors near the left and right edges of the plot correspond to predictions near the interface. E_k^+ and E_k^- decrease with increasing N_{train}.

from predictions near the interfaces and the predictions within the bulk. The interface set I is defined as a set of 100 predictions made by the surrogate near the 2 interfaces. For cations, $I = \{1, 2, 3 ... 50, 452, 453 ... 502\}$, and the corresponding bulk set $B = \{k \in K^+ : k \notin I\}$; for anions $I = \{503, 504 ... 553, 954, 955 ... 1004\}$, and the corresponding bulk set $B = \{k \in K^- : k \notin I\}$. This enables us to define RMSE values E_I^+, E_B^+, E_I^-, and E_B^- associated with interface and bulk for cation and anion density predictions as:

$$E_I^+ = \frac{1}{N_I} \sum_{k \in I} E_k^+, \quad E_B^+ = \frac{1}{N_B} \sum_{k \in B} E_k^+ \tag{6}$$

$$E_I^- = \frac{1}{N_I} \sum_{k \in I} E_k^-, \quad E_B^- = \frac{1}{N_B} \sum_{k \in B} E_k^- \tag{7}$$

Figure 8 shows a bar chart of these interface and bulk RMSE values incurred in surrogate predictions for training dataset size $N_{\text{train}} = 200, 400, 550, 1550, 2550, 3550$. Regardless of the training dataset size, errors incurred in predicting output features associated with the interface are higher than those incurred in predicting output features associated with the bulk. This suggests

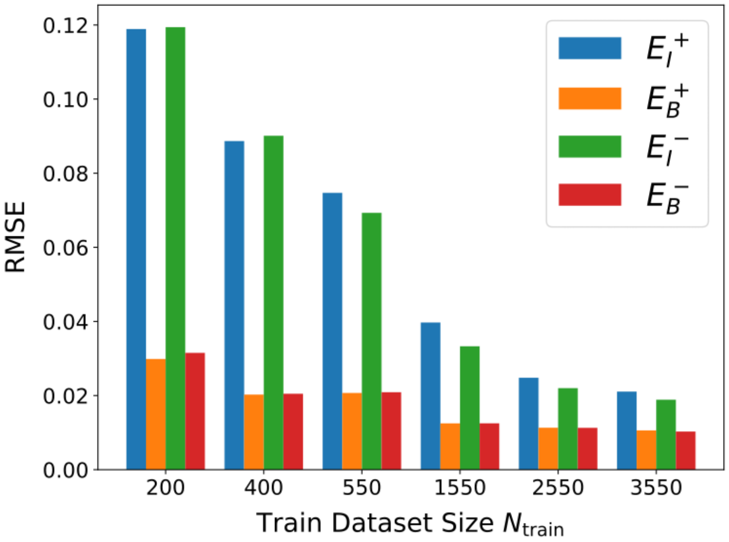

Figure 8: Interface RMSE E_I^+ (blue) and bulk RMSE E_B^+ (orange) for the cation density predictions, and interface RMSE E_I^- (green) and bulk RMSE E_B^- (red) for the anion density predictions vs training dataset size $N_{\text{train}} = 200, 400, 550, 1550, 2550, 3550$. For all N_{train}, interface RMSE $E_I^{+/-}$ are higher than the bulk RMSE $E_B^{+/-}$. All errors are in units of M.

that the model predictions for the interfacial ionic structure are inferior compared to the bulk predictions. The RMSE values E_I^+ and E_I^- associated with the predictions near the interface decrease sharply as N_{train} increases from 200 to 1550. As N_{train} is increased further, these errors decrease relatively slowly, indicating the onset of convergence for $N_{\text{train}} \gtrsim 1550$. The bulk RMSE values E_B^+ and E_B^- decrease as N_{train} increases from 200 to 1550. Further increase in N_{train} only leads to a very slight drop in E_B^+ and E_B^-, indicating convergence.

This analysis suggests that some output features (e.g., interfacial structure) present a more challenging test to the generalization capabilities of the surrogate compared to others (e.g., bulk structure). In addition to assessing the error (E) associated with the overall output, it is thus important to examine the output-specific errors $(E_I^+, E_I^-, E_B^+, E_B^-)$ to credibly assess the convergence and the acceptable level of scientific performance of the surrogate. For the electrolyte system under study, judging by the RMSE values E, E_I^+, E_I^-, E_B^+ and E_B^- as well as the predicted density profiles (Figure 6), an acceptable level of scientific performance for the surrogate at the highest speedup $\mathcal{S} \approx 4/3\mathcal{S}_{\text{HPC}}$ is reached for the training dataset of size $N_{\text{train}} = 1550$. As we move to the studies in Section 3.3, it will be useful to define a reference error scale to judge the surrogate performance. We choose this to be the errors E, E_I^+, E_I^-, E_B^+ and E_B^- associated with the predictions made by the surrogate trained with $N_{\text{train}} = 1550$ samples (e.g., $E \approx 0.017$, $E_I^+ \approx 0.04$ and so on).

3.3 Dataset Composition and Surrogate Generalizability

It is important to recognize that while the qualitative trends shown above regarding the accuracy-speedup tradeoff are expected to hold more generally, the quantitative results regarding the optimal training dataset size are intricately linked to the specific dataset composition. Table 1 highlights the input design space, which shows that different input parameters have different discretizations and ranges. While the ranges get normalized to $(0, 1)$ for all input variables during preprocessing, the different number of discretizations yield differences in the contributions of the input variables toward the learning of the surrogate. We used a dataset of 4050 samples formed via a specific representation of the input design space: $n_h \times n_c \times n_{d_+} \times n_{d_-} \times n_{\sigma_s} \equiv 6 \times 9 \times 5 \times 5 \times 3$, where $n_h, n_c, n_{d_+}, n_{d_-}, n_{\sigma_s}$ are the number of discretizations associated with the input variables h, c, d_+, d_-, σ_s respectively. A different combination of these discretizations yielding 4050 samples will lead to quantitative differences in the RMSE values and the optimal training dataset size.

The specific set of discretizations employed here are the result of a design choice informed by domain knowledge[19,37] and constraints due to the limited computing resources. For example, our recent study[37] on coarse-grained simulations of dense electrolytes was inspired by experiments reporting dramatic changes in the screening behavior of electrolytes with increasing concentration.[38] Thus, the electrolyte concentration c emerged as a key input variable to probe the ionic structure, and is therefore discretized with the most number of values. We also showed that the rise in the steric ion-ion correlations, which depend on the cation size d_+ and anion size d_-, is critical to changes in the ionic structure, particularly in the interfacial regions. This paved the way for selecting a good representation of ion diameters in the discretized input design space. Such domain knowledge infusion is essential for building ML surrogates for MD simulations of soft matter.

In all our previous experiments, the surrogate performance is tested on electrolyte systems in the test dataset S_{test} that the surrogate did not see during training. However, it is very likely that the surrogate encountered the input variables associated with these samples in other combinations. For example, while the surrogate did not see the specific electrolyte system II ($h = 4.4, c = 1.0, d_+ = 0.3075, d_- = 0.63, \sigma_s = -0.01$) during training, it was trained on many systems that have the concentration $c = 1$ M such as the electrolyte system ($h = 4.6, c = 1.0, d_+ = 0.415, d_- = 0.415, \sigma_s = -0.02$). In order to assess the generalizability of the surrogate, it is important to explore its performance on completely unseen input variables.

A campaign that accomplishes this task can be initiated by utilizing the total dataset of 4050 simulations to design training and testing datasets that enable the study of surrogate performance on input variable values obtained via interpolation between the values seen during training. The deterministic separation method (Section 2.3), where pre-selected input variable values are held in a test set hidden from the surrogate training, is suited for this purpose. One can also start a campaign where datasets are designed to enable the study of surrogate performance on input variable values extrapolated outside the region of the input design space. At this time we do not carry out this exercise. Our expectation is that the relatively simple neural network architecture with 2 hidden layers will not fare well on extrapolations.

Surrogate generalizability is key to understanding the potential of computational performance enhancement: a greater degree of generalizability ensures that the surrogate can make a large number of predictions beyond the initial dataset composed of the training and testing samples, thus boosting the speedup obtained in Equation 5. Generalizability is linked to the discretization errors associated with the coarse-grained dataset representative of the continuum input material design space, and to challenges associated with capturing a specific feature in the output which may correlate strongly to one or more input variables. For example, interfacial ionic structure is strongly correlated with the ion size. To assess surrogate generalizability, we carry out 3 studies that involve determining the surrogate performance on unseen input variable values obtained via interpolation between the seen ones. In all these studies, the surrogate is trained using the deterministic separation method outlined in Section 2.2. The validation loss is observed to decrease with increasing number of epochs in all cases, yielding convergence for $n_e > 15000$. The optimal

ANN models are built by checkpointing at $n_e = 20000$.

We begin with excluding systems characterized with electrolyte concentration $c = 1$ M from the training dataset S_{train}, and using these excluded systems to create the test dataset S_{test}. In order to study the dependence of the surrogate performance on the number of $c = 1$ M samples the surrogate sees during training, we define $f = 100A_{\text{trans}}/N_{\text{test}}$ as the percentage of test samples appended to the training dataset S_{train}, where A_{trans} denotes the number of samples drawn randomly from the test set. $f = 0\%$ means S_{test} contains all electrolyte systems with $c = 1$ M, and S_{train} contains none, signaling that the surrogate will make predictions in a completely "blind" mode. $f = 50\%$ implies 50% of samples from S_{test} are randomly drawn and appended to S_{train}, which are likely to "informate" the surrogate learning of the features associated with the hidden, interpolated input variable value. The training and testing dataset sizes $(N_{\text{train}}, N_{\text{test}})$ for $f = 0\%, 1\%, 10\%$ and 50% are: $(3600, 450), (3604, 446), (3645, 405), (3825, 225)$ respectively. These different dataset compositions enable the probing of the generalization ability of the surrogate on unseen $c = 1$ M electrolyte systems after it sees $A_{\text{trans}} = 0, 4, 45$, and 225 samples characterized with $c = 1$ M.

Figure 9(a) shows a bar chart of the average interface and bulk RMSE values for cations (E_I^+, E_B^+) and anions (E_I^-, E_B^-) as a function of f. Figure 9(b) shows the cation density profiles predicted by the surrogate for a representative system in the test set characterized with the input variable combination of $(4.2, 1.0, 0.415, 0.415, -0.01)$. The $f = 0\%$ result exhibits very large errors $E_I^+ \approx 0.125$, $E_I^- \approx 0.1$, and $E_B^{+/-} \approx 0.175$, which shows that the model fails to generalize well on this interpolated input variable value if it does not "see" any $c = 1$ M samples. The corresponding density profile result completely misses the ground truth.

For $f = 1\%$, which corresponds to only four $c = 1$ M samples seen by the surrogate, the model gains knowledge and adjusts its weights and biases, yielding $3\times$ smaller RMSE values and a much improved density profile prediction. The interface RMSE values $E_I^{+/-}$ approach the acceptable reference error scale (≈ 0.04) set by the errors associated with the training dataset of 1550 samples (Section 3.2). However, the average RMSE is higher than the reference value ($E \approx 0.017$) due to the relatively high bulk RMSE values $\sim E_B^{+/-} \approx 0.05$. For $f = 10\%$, which corresponds to the

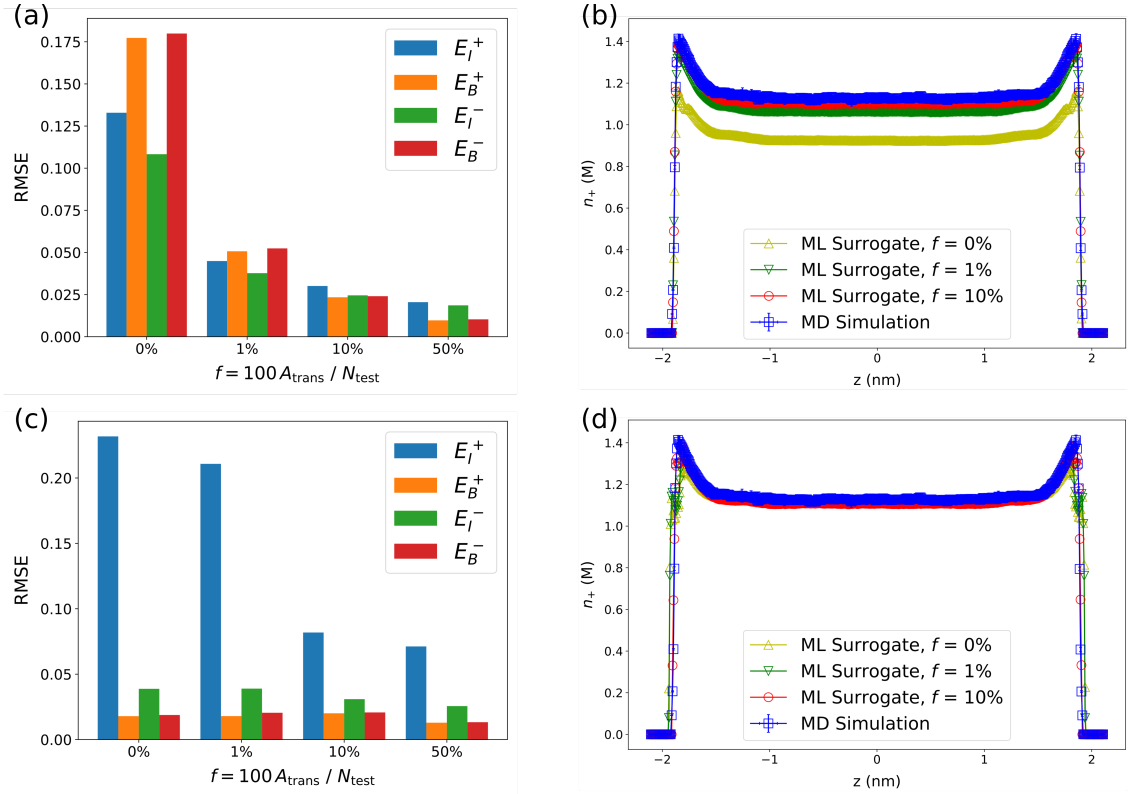

Figure 9: RMSE values (in units of M) and cation density profiles associated with the predictions made by the surrogate trained using datasets generated via the deterministic separation method by excluding electrolyte concentration $c = 1$ M (a, b) or by excluding the cation diameter $d_+ = 0.415$ nm (c, d). (a) and (d) show interface RMSE E_I^+ (blue) and bulk RMSE E_B^+ (orange) for the cation density predictions, and interface RMSE E_I^- (green) and bulk RMSE E_B^- (red) for the anion density predictions for different percentages $f = 0\%, 1\%, 10\%, 50\%$ of samples appended to the train dataset. (b) and (d) show cation density profiles for the same electrolyte system in the test set characterized with input variables $(4.2, 1.0, 0.415, 0.415, -0.01)$. Yellow up triangles, green down triangles, red circles represent surrogate predictions with $f = 0\%, 1\%, 10\%$ respectively. Blue squares with errorbars show the ground truth results produced by MD simulations.

surrogate seeing 45 systems with $c = 1$ M, $E_I^{+/-} \sim E_B^{+/-} \approx 0.025$. While the bulk RMSE values are still on the higher side, the overall error is low and close to the reference RMSE E. Further, the corresponding density profile prediction agrees well with the MD simulation results, particularly near the interfaces. Errors $E_I^{+/-}$ and $E_B^{+/-}$ associated with $f = 50\%$ are smaller than the reference errors, indicating the convergence of the surrogate accuracy.

We next perform a similar study by hiding the cation diameter value $d_+ = 0.415$ nm during the training of the surrogate. The training and testing datasets associated with $f = 0\%, 1\%, 10\%,$

and 50% are $(3240, 810), (3248, 802), (3321, 729)$, and $(3645, 405)$ respectively. These different dataset compositions enable the probing of the generalization ability of the surrogate on unseen electrolyte systems with cations of diameter $d_+ = 0.415$ nm, after the surrogate sees $A_{\text{trans}} = 0, 8, 81$, and 405 systems with cations of diameter $d_+ = 0.415$ nm. Figure 9(c) and 9(d), respectively, show the RMSE errors $E_I^+, E_B^+, E_I^-, E_B^-$, and the predicted cation density profiles for the same electrolyte system used in the previous study. A very different picture emerges in comparison to the study depicted in Figures 9(a) and 9(b), where the pre-selected hidden input variable is $c = 1$ M. For all f values, $E_B^{+/-} < 0.02$ and the predicted density profiles show that the surrogate generalizes well for the bulk region, even for $f = 0\%$. $E_I^- \approx 0.04$ for anions at $f = 0\%$ is close to the acceptable reference error scale, and is reduced by half for $f = 10\%$. The corresponding surrogate predictions for the anion density profiles agree well with the ground truth.

On the other hand, the interface RMSE values for cations start out $5\times$ bigger than E_I^- at $f = 0\%$ and do not decrease sharply with increasing f, dropping to $E_I^+ \approx 0.07$ at $f = 50\%$. The predicted cation density profiles are consistent with these errors, indicating that the prediction of the cation density profile near the interface is challenging for the surrogate if it does not "see" the cation diameter during training. This is consistent with our physical understanding that the cation size is the primary determinant of the cation contact density near the confining surfaces.[37] In another study, an anion diameter value was excluded from the surrogate training, and we found analogous results: predictions were significantly poorer for the anion density profile near the interfaces, while other output features were predicted with acceptable accuracy.

We now perform a study to assess the surrogate generalizability in making predictions for multiple electrolyte concentrations interpolated between the seen values. Electrolyte concentrations $c = 0.25, 0.75, 1.25, 1.75$ M are excluded from the training dataset and the surrogate is trained on $c = 0.1, 0.5, 1.0, 1.5, 2.0$ M. This effectively increases the discretization step by a factor of ≈ 2. The surrogate is then tasked to make predictions for the interpolated c values, i.e., $c = 0.25, 0.75, 1.25, 1.75$ M in the test dataset. We generate the training and testing datasets for $f = 0\%$, 1%, 10%, and 50%, whose sizes $(N_{\text{train}}, N_{\text{test}})$ are $(2250, 1800), (2268, 1782), (2430, 1620)$,

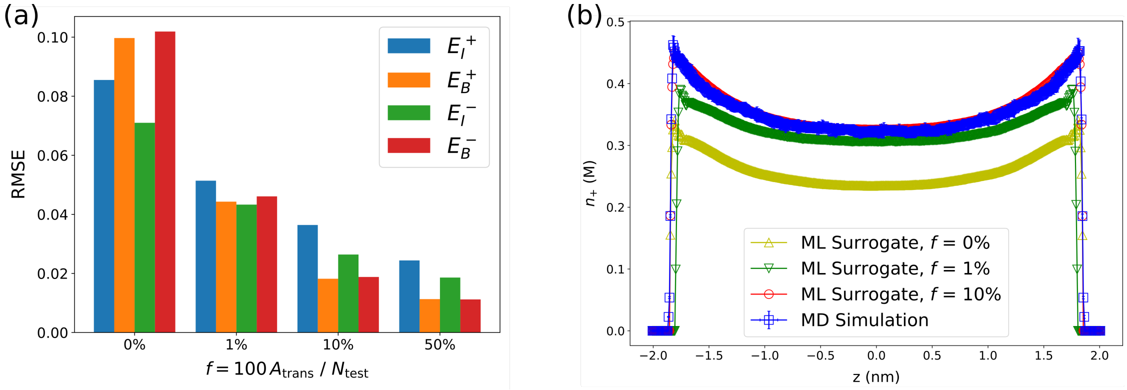

Figure 10: RMSE values in units of M (a) and cation density profiles (b) associated with the predictions made by the surrogate trained using datasets generated by the deterministic separation method via excluding electrolyte concentration $c = 0.25, 0.75, 1.25, 1.75$ M. (a) Interface RMSE E_I^+ (blue) and bulk RMSE E_B^+ (orange) for the cation density predictions, and interface RMSE E_I^- (green) and bulk RMSE E_B^- (red) for the anion density predictions decrease with an increase in the percentage f of samples removed from the test dataset and appended to the train dataset. (b) Yellow up triangles, green down triangles, red circles represent surrogate predictions with $f = 0\%, 1\%, 10\%$ respectively for an electrolyte system in the test set characterized with input variables $(4, 0.25, 0.3075, 0.415, -0.015)$. With increasing f, surrogate predictions get closer to the ground truth results (blue squares) produced by MD simulations.

and $(3150, 900)$ respectively. Using these different dataset compositions, we evaluate the surrogate performance on electrolyte systems characterized with $c = 0.25, 0.75, 1.25, 1.75$ M after it sees $A_{\text{trans}} = 0, 18, 180$, and 900 samples with these concentrations.

Figure 10(a) shows a bar chart of the resulting RMSE values E_I^+, E_B^+, E_I^-, E_B^- and Figure 10(b) shows the predicted density profiles for a representative system in the test dataset characterized with input variables $(4, 0.25, 0.3075, 0.415, -0.015)$. For $f = 0\%$, the model predictions incur large interface and bulk errors, and the associated density profile misses the ground truth entirely, indicating the inability of the surrogate to generalize without seeing any systems characterized with the interpolated c values. The $f = 1\%$ result shows improvement, but the errors are still large and the agreement with the ground truth is poor. For $f = 10\%$, the errors are below or close to the acceptable reference errors, and the predicted density profile exhibits a good agreement with the ground truth result. The surrogate accuracy converges for $f = 50\%$ as evident by the errors E_I^+, E_B^+, E_I^-, E_B^- well below the reference values.

The last study sheds light on the accuracy-speedup tradeoff probed in Section 3.1. Firstly, the surrogate does not need to see all possible combinations of the interpolated values with the other input variables in order to achieve an acceptable level of scientific performance. To illustrate, starting from a dataset of 2250 samples ($f = 0\%$), for which the surrogate performs poorly, we only need to add up to 900 samples ($f = 50\%$) in order to achieve acceptable accuracy. In other words, instead of a dataset containing 4050 samples obtained with running simulations on all the finer-resolution grid points generated via interpolation, the target accuracy can be achieved with a dataset of less than 3150 samples. This reduction in the training dataset size increases the speedup.

A simple estimate of a speedup achieved through this interpolation approach can be derived by utilizing the f parameter, which represents the fraction of the new systems a surrogate is shown before it is tasked to make predictions. Before interpolation, the potential gains possible with the surrogate application is captured by the the baseline speedup $\mathcal{S}_B$ given as

$$\mathcal{S}_B = \mathcal{S}_{\text{HPC}} \frac{N_{\text{total}}}{N_{\text{train}}}. \tag{8}$$

This equation is similar to Equation 5 except that we assume the surrogate can make N_{total} predictions at an acceptable level of accuracy, where N_{total} is the total number of samples in the dataset.

For simplicity, we consider the case of interpolating on one input variable (dimension). As the discretization step is reduced by half, the number of predictions approximately doubles to $2N_{\text{total}}$. However, the number of training samples required to get a well-trained surrogate increases to $N_{\text{train}} + fN_{\text{total}}$. For instance, in the above study, $f \in (10, 50)\%$. Following the process used in defining the baseline speedup, the net speedup $\mathcal{S}'$ associated with the surrogate performance considering both the pre-interpolation and the post-interpolation phases can be written as:

$$\mathcal{S}' \approx \mathcal{S}_{\text{HPC}} \frac{2N_{\text{total}}}{N_{\text{train}} + fN_{\text{total}}}. \tag{9}$$

The net speedup $\mathcal{S}'$ decreases as f increases. In other words, if the surrogate needs to see a large fraction of the interpolated values in order to make predictions at the accepted accuracy level, then the speedup will be small. In another scenario, if one can tolerate larger errors incurred in the predictions made by a surrogate trained on a small fraction of the interpolated values, then the speedup can be boosted. This is another manifestation of the accuracy-speedup tradeoff.

By replacing $N_{\text{total}}/N_{\text{train}}$ in Equation 9 with $\mathcal{S}_B/\mathcal{S}_{\text{HPC}}$ using Equation 8, $\mathcal{S}'$ can be expressed as

$$\mathcal{S}' \approx \frac{2\mathcal{S}_B\mathcal{S}_{\text{HPC}}}{\mathcal{S}_{\text{HPC}} + f\mathcal{S}_B} = \frac{2}{1/\mathcal{S}_B + f/\mathcal{S}_{\text{HPC}}}. \tag{10}$$

A number of qualitative insights follow from Equation 10 regarding the potential gains associated with the application of the surrogate via the interpolation approach to generate more predictions. If the interpolation is such that the surrogate needs to see all $(f = 100\%)$ of the new potential predictions (interpolated samples) in order to achieve the acceptable accuracy, then the net speedup $\mathcal{S}'$ is bounded from above by $2\mathcal{S}_{\text{HPC}}$, which is the limit of taking the baseline speedup $\mathcal{S}_B \to \infty$. For this case, the lower bound of $\mathcal{S}'$ is $\mathcal{S}_{\text{HPC}}$, which is the same as that of $\mathcal{S}_B$. These lower and upper bounds of $\mathcal{S}'$ will increase as f decreases. For example, if the surrogate only needs to see half $(f = 50\%)$ of the new samples obtained via interpolation to make predictions with acceptable accuracy, the net speedup is bounded by $(4/3)\mathcal{S}_{\text{HPC}} < \mathcal{S}' < 4\mathcal{S}_{\text{HPC}}$. The maximum possible net speedup is doubled compared to the case where the surrogate needs to see all of the new samples. An interesting possibility arises for $f = 0\%$, which indicates that the baseline surrogate is already well generalized and will predict with acceptable accuracy on all new samples generated via interpolation. For this case, we get $\mathcal{S}' = 2\mathcal{S}_B > 2\mathcal{S}_{\text{HPC}}$, i.e., the net speedup scales linearly with the baseline speedup, and while it has a lower bound of $2\mathcal{S}_{\text{HPC}}$, it does not have an upper bound.

4 Conclusions

We have conducted a systematic study of the tradeoff between the scientific and the computational performance associated with ML surrogates for MD simulations of soft materials. The study used a dataset generated by conducting simulations of 4050 different electrolyte systems that exhibit a rich and complex relationship between the input electrolyte attributes and the output ionic structure. The surrogate was tasked to learn the relationship between 1004 output features characterizing the ionic distributions and 5 input features describing the electrolyte system: confinement length, electrolyte concentration, cation diameter, anion diameter, and surface charge density.

The scientific performance or accuracy was measured by computing RMSE values between the surrogate predictions and the ground truth results obtained via MD simulations, as well as by comparing the output features obtained via the two approaches. The computational performance was evaluated by computing the speedup which incorporated the training dataset creation time. A power-law decrease in the overall RMSE was observed with increasing training dataset size $N_{\text{train}} \in (150, 3550)$, with the onset of convergence for $N_{\text{train}} \gtrsim 1550$ samples. This improvement in the prediction accuracy with increasing N_{train} was accompanied by a reduction in the speedup.

A comprehensive assessment of the scientific performance was obtained by evaluating the output-feature-specific surrogate accuracies via the computation of the RMSE values associated with the interfacial and bulk regions separately for cations and anions. Predicting output features associated with the interfacial regions incurred larger errors compared to the features associated with the bulk regions. An acceptable level of accuracy was reached for the training dataset with $N_{\text{train}} = 1550$ samples based on the overall and output-specific RMSE values and the agreement between the predicted density profiles and the ground truth. This training dataset was found to be optimal under the constraint of maximizing the speedup.

The generalizability of the surrogate was explored by testing its performance on unseen values of the input variables obtained via interpolation. The surrogate performance was affected by which input variable (material attribute) was hidden. Showing larger fractions of the new interpolated samples to the surrogate during training improved its accuracy, but at the cost of reducing the potential of computational gains. This tradeoff was captured by developing a net speedup metric that revealed qualitative insights about the bounds on the computational gains associated with the surrogate if the interpolation approach is adopted to generate new predictions.

The interpolation study shows that the brute force approach of reducing the discretization step of the input variables to generate simulations for a larger set of grid points is not only computationally prohibitive, it may not yield substantial improvements in surrogate accuracy as the latter converges with far fewer samples. Further, the surrogate does not need to see all possible combinations of the interpolated input variable value with the other input variables in order to achieve an acceptable accuracy level. Eliminating the unnecessary simulations to further reduce training dataset size may require the use of active learning based methods to crawl through the input design space. Exploring such smart sampling methods[39,40] to determine the training datasets of optimal size and composition will be a subject of future work.

At present, the surrogate design and MD simulations employ different environments and workflows. Recent work has investigated the use of ML platforms to improve the execution (e.g., accuracy, performance) of MD simulations themselves.[41–45] Our future work will leverage these ideas to explore simplifying the end-to-end surrogate design process by developing a unified framework to enable the execution of MD simulations and surrogate design tasks in a one-stop platform.[45]

ACKNOWLEDGMENTS

Funding from the National Science Foundation through Awards 1720625 and DMR-1753182, and from the Department of Energy through Award DE-SC0023452 is gratefully acknowledged. Simulations were performed using the BigRed3 supercomputing system at Indiana University.

References

(1) Ferguson, A. L. Machine learning and data science in soft materials engineering. Journal of Physics: Condensed Matter **2017**, 30, 043002.

(2) Spellings, M.; Glotzer, S. C. Machine learning for crystal identification and discovery. AIChE Journal **2018**, 64, 2198–2206.

(3) Schoenholz, S. S.; Cubuk, E. D.; Kaxiras, E.; Liu, A. J. Relationship between local structure and relaxation in out-of-equilibrium glassy systems. Proceedings of the National Academy of Sciences **2017**, 114, 263–267.

(4) Guo, A. Z.; Sevgen, E.; Sidky, H.; Whitmer, J. K.; Hubbell, J. A.; de Pablo, J. J. Adaptive enhanced sampling by force-biasing using neural networks. The Journal of chemical physics **2018**, 148, 134108.

(5) Wessels, M. G.; Jayaraman, A. Machine Learning Enhanced Computational Reverse Engineering Analysis for Scattering Experiments (CREASE) to Determine Structures in Amphiphilic Polymer Solutions. ACS Polymers Au **2021**, 8581–8591.

(6) Wang, J.; Olsson, S.; Wehmeyer, C.; Perez, A.; Charron, N. E.; De Fabritiis, G.; Noe, F.; Clementi, C. Machine learning of coarse-grained molecular dynamics force fields. ACS central science **2019**,

(7) Casalino, L.; Dommer, A. C.; Gaieb, Z.; Barros, E. P.; Sztain, T.; Ahn, S.-H.; Trifan, A.; Brace, A.; Bogetti, A. T.; Clyde, A.; Ma, H.; Lee, H.; Turilli, M.; Khalid, S.; Chong, L. T.; Simmerling, C.; Hardy, D. J.; Maia, J. D.; Phillips, J. C.; Kurth, T.; Stern, A. C.; Huang, L.; McCalpin, J. D.; Tatineni, M.; Gibbs, T.; Stone, J. E.; Jha, S.; Ramanathan, A.; Amaro, R. E. AI-driven multiscale simulations illuminate mechanisms of SARS-CoV-2 spike dynamics. The International Journal of High Performance Computing Applications **2021**, 35, 432–451.

(8) Kadupitiya, J.; Jadhao, V. Probing the Rheological Properties of Liquids Under Conditions of Elastohydrodynamic Lubrication Using Simulations and Machine Learning. Tribology Letters **2021**, 69, 1–19.

(9) Moradzadeh, A.; Aluru, N. R. Molecular Dynamics Properties without the Full Trajectory: A Denoising Autoencoder Network for Properties of Simple Liquids. The journal of physical chemistry letters **2019**, 10, 7568–7576.

(10) Häse, F.; Fdez. Galván, I.; Aspuru-Guzik, A.; Lindh, R.; Vacher, M. How machine learning can assist the interpretation of ab initio molecular dynamics simulations and conceptual understanding of chemistry. Chem. Sci. **2019**, 10, 2298–2307.

(11) Sun, Y.; DeJaco, R. F.; Siepmann, J. I. Deep neural network learning of complex binary sorption equilibria from molecular simulation data. Chemical science **2019**, 10, 4377–4388.

(12) Kasim, M. F.; Watson-Parris, D.; Deaconu, L.; Oliver, S.; Hatfield, P.; Froula, D. H.; Gregori, G.; Jarvis, M.; Khatiwala, S.; Korenaga, J.; Topp-Mugglestone, J.; Viezzer, E.; Vinko, S. M. Building high accuracy emulators for scientific simulations with deep neural architecture search. Machine Learning: Science and Technology **2021**, 3, 015013.

(13) Kadupitiya, J.; Fox, G. C.; Jadhao, V. Machine learning for parameter auto-tuning in molecular dynamics simulations: Efficient dynamics of ions near polarizable nanoparticles. The International Journal of High Performance Computing Applications **2020**, 34, 357–374.

(14) Kadupitiya, J.; Fox, G. C.; Jadhao, V. Solving Newton's equations of motion with large timesteps using recurrent neural networks based operators. Machine Learning: Science and Technology **2022**, 3, 025002.

(15) Kadupitiya, J.; Fox, G. C.; Jadhao, V. Machine learning for performance enhancement of molecular dynamics simulations. International Conference on Computational Science. 2019; pp 116–130.

(16) Kadupitiya, J.; Sun, F.; Fox, G. C.; Jadhao, V. Machine learning surrogates for molecular dynamics simulations of soft materials. Journal of Computational Science **2020**, 42, 101107.

(17) Degiacomi, M. T. Coupling molecular dynamics and deep learning to mine protein conformational space. Structure **2019**, 27, 1034–1040.

(18) Jadhao, V.; Kadupitiya, J. Integrating Machine Learning with HPC-driven Simulations for Enhanced Student Learning. 2020 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC). Los Alamitos, CA, USA, 2020; pp 25–34.

(19) Jing, Y.; Jadhao, V.; Zwanikken, J. W.; Olvera de la Cruz, M. Ionic structure in liquids confined by dielectric interfaces. The Journal of chemical physics **2015**, 143, 194508.

(20) He, Y.; Gillespie, D.; Boda, D.; Vlassiouk, I.; Eisenberg, R. S.; Siwy, Z. S. Tuning transport properties of nanofluidic devices with local charge inversion. Journal of the American Chemical Society **2009**, 131, 5194–5202.

(21) Faucher, S.; Aluru, N.; Bazant, M. Z.; Blankschtein, D.; Brozena, A. H.; Cumings, J.; Pedro de Souza, J.; Elimelech, M.; Epsztein, R.; Fourkas, J. T.; Rajan, A. G.; Kulik, H. J.; Levy, A.; Majumdar, A.; Martin, C.; McEldrew, M.; Misra, R. P.; Noy, A.; Pham, T. A.; Reed, M.; Schwegler, E.; Siwy, Z.; Wang, Y.; Strano, M. Critical Knowledge Gaps in Mass Transport through Single-Digit Nanopores: A Review and Perspective. The Journal of Physical Chemistry C **2019**, 123, 21309–21326.

(22) Park, H. B.; Kamcev, J.; Robeson, L. M.; Elimelech, M.; Freeman, B. D. Maximizing the right stuff: The trade-off between membrane permeability and selectivity. Science **2017**, 356, eaab0530.

(23) Werber, J. R.; Osuji, C. O.; Elimelech, M. Materials for next-generation desalination and water purification membranes. Nature Reviews Materials **2016**, 1, 1–15.

(24) Levin, Y. Strange electrostatics in physics, chemistry, and biology. Physica A: Statistical Mechanics and its Applications **2005**, 352, 43 – 52.

(25) Zwanikken, J. W.; Olvera de la Cruz, M. Correlated electrolyte solutions and ion-induced attractions between nanoparticles. Phys. Rev. E **2010**, 82, 050401.

(26) Raudys, S.; Jain, A. Small sample size effects in statistical pattern recognition: recommendations for practitioners. IEEE Transactions on Pattern Analysis and Machine Intelligence **1991**, 13, 252–264.

(27) Jain, A. K.; Chandrasekaran, B. 39 Dimensionality and sample size considerations in pattern recognition practice. Handbook of statistics **1982**, 2, 835–855.

(28) Park, S. H.; Han, K. Methodologic guide for evaluating clinical performance and effect of artificial intelligence technology for medical diagnosis and prediction. Radiology **2018**, 286, 800–809.

(29) Schnack, H. G.; Kahn, R. S. Detecting neuroimaging biomarkers for psychiatric disorders: sample size matters. Frontiers in psychiatry **2016**, 7, 50.

(30) Schmidt, J.; Shi, J.; Borlido, P.; Chen, L.; Botti, S.; Marques, M. A. Predicting the thermodynamic stability of solids combining density functional theory and machine learning. Chemistry of Materials **2017**, 29, 5090–5103.

(31) Boda, D.; Gillespie, D.; Nonner, W.; Henderson, D.; Eisenberg, B. Computing induced charges in inhomogeneous dielectric media: Application in a Monte Carlo simulation of complex ionic systems. Phys. Rev. E **2004**, 69, 046702.

(32) Allen, R.; Hansen, J.-P.; Melchionna, S. Electrostatic potential inside ionic solutions confined by dielectrics: a variational approach. Phys. Chem. Chem. Phys. **2001**, 3, 4177–4186.

(33) Tyagi, S.; Suzen, M.; Sega, M.; Barbosa, M.; Kantorovich, S. S.; Holm, C. An iterative, fast, linear-scaling method for computing induced charges on arbitrary dielectric boundaries. The Journal of Chemical Physics **2010**, 132, 154112.

(34) Barros, K.; Sinkovits, D.; Luijten, E. Efficient and accurate simulation of dynamic dielectric objects. The Journal of Chemical Physics **2014**, 140, 064903.

(35) Plimpton, S. Fast Parallel Algorithms for Short-Range Molecular Dynamics. Journal of Computational Physics **1995**, 117, 1 – 19.

(36) Deserno, M.; Holm, C. How to mesh up Ewald sums. I. A theoretical and numerical comparison of various particle mesh routines. The Journal of chemical physics **1998**, 109, 7678–7693.

(37) Anousheh, N.; Solis, F. J.; Jadhao, V. Ionic structure and decay length in highly concentrated confined electrolytes. AIP Advances **2020**, 10, 125312.

(38) Smith, A. M.; Lee, A. A.; Perkin, S. The electrostatic screening length in concentrated electrolytes increases with concentration. The journal of physical chemistry letters **2016**, 7, 2157–2163.

(39) Pestourie, R.; Mroueh, Y.; Nguyen, T. V.; Das, P.; Johnson, S. G. Active learning of deep surrogates for PDEs: application to metasurface design. npj Computational Materials **2020**, 6, 1–7.

(40) Lee, H.; Turilli, M.; Jha, S.; Bhowmik, D.; Ma, H.; Ramanathan, A. Deepdrivemd: Deep-learning driven adaptive molecular simulations for protein folding. 2019 IEEE/ACM Third Workshop on Deep Learning on Supercomputers (DLS). 2019; pp 12–19.

(41) Yao, K.; Herr, J. E.; Toth, D. W.; Mckintyre, R.; Parkhill, J. The TensorMol-0.1 model chemistry: a neural network augmented with long-range physics. Chemical science **2018**, 9, 2261–2269.

(42) Gao, X.; Ramezanghorbani, F.; Isayev, O.; Smith, J. S.; Roitberg, A. E. TorchANI: A free and open source PyTorch-based deep learning implementation of the ANI neural network potentials. Journal of chemical information and modeling **2020**, 60, 3408–3415.

(43) Doerr, S.; Majewski, M.; Pérez, A.; Krämer, A.; Clementi, C.; Noe, F.; Giorgino, T.; De Fabritiis, G. TorchMD: A Deep Learning Framework for Molecular Simulations. Journal of Chemical Theory and Computation **0**, 0, null.

(44) Barrett, R.; Chakraborty, M.; Amirkulova, D. B.; Gandhi, H. A.; Wellawatte, G. P.; White, A. D. Hoomd-tf: Gpu-accelerated, online machine learning in the hoomd-blue molecular dynamics engine. Journal of Open Source Software **2020**, 5, 2367.

(45) Sharma, P.; Jadhao, V. Molecular Dynamics Simulations on Cloud Computing and Machine Learning Platforms. 2021 IEEE 14th International Conference on Cloud Computing (CLOUD). Los Alamitos, CA, USA, 2021; pp 751–753.