Machine learning surrogates for molecular dynamics simulations of soft materials

JCS Kadupitiya[a], Fanbo Sun[a], Geoffrey Fox[a] and Vikram Jadhao[a,*]

[a] *Intelligent Systems Engineering, Indiana University, Bloomington, Indiana 47408*

ARTICLE INFO

Keywords:
Machine Learning
Molecular Dynamics Simulations
Parallel Computing
Scientific Computing
Soft Materials

ABSTRACT

Molecular dynamics (MD) simulations accelerated by high-performance computing (HPC) methods are powerful tools to investigate and extract the microscopic mechanisms characterizing the properties of soft materials such as self-assembled nanoparticles, virus capsids, confined electrolytes, and polymeric fluids. In this paper, we extend the idea, first proposed in Ref. [36], of integrating machine learning (ML) methods with HPC-accelerated MD simulations of soft materials to enhance their predictive power and advance their applications for research and educational activities. The idea is illustrated using parallelized MD simulations designed to extract the distribution of self-assembling ions in nanoconfinement. We find that an artificial neural network-based regression model successfully learns nearly all the interesting features associated with the output ionic density profiles over a broad range of ionic system parameters. The ML model generates predictions that are in excellent agreement with the results from MD simulations. The inference time associated with the ML model is over a factor of 10,000 smaller than the corresponding parallel MD simulation time. Through this demonstration, we introduce a “machine learning surrogate” for MD simulations of soft-matter systems. To realize the advantages associated with the ML surrogate, we develop a nanoHUB web application based on the initial proposal outlined in Ref. [36]. The results demonstrate that the performance of MD simulations can be further enhanced by using ML, enabling rapid and accurate simulation-driven exploration of soft material design space.

1. Introduction

Molecular dynamics simulations are powerful tools for investigating the microscopic origins of the behavior of materials including soft matter. These simulations have enabled the understanding of microscopic mechanisms underlying the assembly of both biological and synthetic soft materials such as virus capsids [47, 20], confined electrolytes [3], polymeric liquids [40, 28], and self-assembled nanostructures [18, 10]. The molecular dynamics (MD) method solves the Newton’s equation of motion for a system of many particles and evolves the positions, velocities, and forces associated with these particles at each time step. While MD simulations are generalizable to study a broad range of phenomena in soft matter, they incur high computational costs because the computational complexity per time step is proportional to the square of the total number of particles.

The high computational costs associated with MD simulations are typically mitigated by employing high-performance computing (HPC) resources and utilizing parallel computing techniques such as OpenMP and MPI. For example, in a typical molecular dynamics (MD) simulation of electrolyte ions confined by material surfaces [3, 33, 38], $\approx$ 1 nanosecond of dynamics of $\approx$ 500 ions on one processor takes $\approx$ 12 hours of runtime. This timescale is prohibitively large to extract converged results for ion distributions that generally require $\approx$ 5 nanoseconds of simulated dynamics. Performing the same simulation on a single node with multiple cores using OpenMP shared memory reduces the runtime by a factor of 10 ($\approx$ 1 hour), enabling simulations of the same system for longer physical times. Using MPI dramatically enhances the simulation performance: for systems with thousands of ions, speedup of over 100 can be achieved, enabling the generation of the needed data for evaluating converged ionic distributions over a broad region of parameters characterizing the ionic system design space. Further, a hybrid OpenMP/MPI approach can provide even higher speedup of over 400 for similar-size systems, enabling MD simulations that can explore the long-time dynamics of a large number of ions with fewer controlled approximations [37].

The HPC-accelerated MD simulations are not only useful for state-of-the-art research, but can be employed as educational tools for teaching related computational science and engineering concepts [23]. However, despite the employment of the optimal parallelization models suited for the size and complexity of the system, scientific simulations can often take hours or days to furnish accurate output data and desired information. To expedite MD simulation-driven design of advanced soft materials, it is desirable to rapidly access trends associated with relevant physical quantities that could be learned and predicted with reasonable accuracy based on the history of data generated from earlier simulation runs. Further, in the area of using simulations in education, rapid access to simulation-driven responses to student questions in classroom settings are desirable in explaining concepts. In the end, there is a critical need for new approaches to accelerate simulations, leverage past simulations to generate accurate predictions, and expedite the analysis of simulation data and classify material properties.

Machine learning (ML) has the potential to directly address the aforementioned critical need. Accordingly, there

[*] Corresponding author
🖂 vjadhao@iu.edu (V. Jadhao)
ORCID(s): 0000-0002-8034-2654 (V. Jadhao)

has been a surge in the use of ML to accelerate computational techniques aimed at understanding material phenomena [16]. ML has been used to predict parameters, generate configurations in material simulations, and classify material properties [52, 49, 42, 45, 13, 5, 7, 43, 16, 19, 37]. Motivated by the advancements in ML and by the challenges in employing MD simulations in research and education, in a recent paper [36], we introduced the idea of integrating ML methods with MD simulations to enhance their performance and overall usability. We demonstrated that an artificial neural network (ANN) based regression model, trained on data generated via HPC-accelerated MD simulations, successfully learns a small number of pre-identified features associated with the simulation output. The ML model instantaneously generated predictions in excellent agreement with results obtained from explicit MD simulations [36].

In this paper, we extend the original idea and work to the more challenging problem of capturing nearly all the interesting features of the desired simulation output. We utilize the MD simulations of ions in nanoconfinement employed in our earlier work [33, 36] to illustrate the results. While the earlier paper showed a relatively small number (3) of ML-generated predictions for the ionic distribution (the contact, mid-point, and peak ionic densities), we now demonstrate that the ANN model trained on the same dataset yields accurate predictions for ≈ 150 output parameters, enabling the estimation of almost all the interesting features of the ionic density profile for a wide range of system parameters. The inference time associated with the ML method is over a factor of $10,000$ smaller than the corresponding MD simulation time. Through this demonstration, we introduce a first-of-its-kind "machine learning surrogate" for MD simulations of soft-matter systems.

These ML surrogates for MD simulations expand the research explorations to a much larger set of model parameters, enabling the rapid identification of interesting regimes and reliable estimates of soft material properties. In addition to expediting research, real-time access to simulation-driven responses to student questions enables a dynamic environment for using simulation for educational activities in classroom settings. To realize these advantages, we integrated the ML surrogate with a web application on nanoHUB: Ions in nanoconfinement. The ML-enhanced GUI delivers a dynamic and responsive environment to users, providing instantaneous and accurate predictions of the ionic density profile over a wide range of ionic attributes.

The organization of the paper is as follows. Sec. 2 provides the background and related work. Sec. 3 describes the ML surrogate approach. Sec. 4 presents the results of the application of the ML surrogate to the example soft-matter framework of self-assembling ions in nanoconfinement. Finally, Sec. 5 provides a brief discussion and summary including our plans for future work. For the sake of clarity and continuity, we review some important findings from the original conference paper [36] as preliminary results in Sec. 4. Similarly, some of the content in background and related work in the original paper is repeated here.

2. Background and Related Work

MD simulations serve as important tools for understanding diverse self-assembly phenomena in nanoscale materials [29, 32, 18], predicting material behavior in practical applications [27], and isolating interesting regions of parameter space for experimental exploration [9]. As in our original conference proceedings [36], we focus specifically on MD simulations of ions in nanoconfinement in this work to illustrate the idea of ML surrogates.

2.1. MD simulations of ions in nanoconfinement

Electrolyte ions drive key processes involved in the generation and function of soft materials such as the morphological changes in proteins, stabilization of colloids, pattern formation in nanostructures, and emulsion-based extraction of metal ions from wastewater. As a result, investigating the self-assembly of ions and extracting their distributions in nanoconfinement created by surfaces of materials such as nanoparticles, colloids, or biological macromolecules has been the focus of many experimental, theoretical, and computational studies [44, 4, 29, 31, 51, 30, 46, 15]. Confined electrolyte solutions themselves are important soft-matter systems that are ubiquitous in biology and modern technological devices such as supercapcitors [3, 44, 2, 50, 33].

From a computational modeling standpoint, the ions are represented as spheres of finite size, the material surfaces are often treated as planar interfaces considering the size difference between the ions and the confining material particles, and the solvent is coarse-grained to speed-up the simulations. Such coarse-grained MD simulations have been employed to extract the ionic distributions over a wide range of electrolyte concentrations, ion valencies, and interfacial separations using codes developed in individual research groups [3, 6, 33] or using general purpose software packages such as ESPRESSO [41] and LAMMPS [48].

2.2. ML-enhanced simulations of materials

Recent years have seen a surge in the use of ML to accelerate computational techniques aimed at understanding material phenomena. ML has been used to predict parameters, generate configurations in material simulations, design coarse-grained potentials, and classify material properties [7, 16, 42, 52, 53, 22]. For example, Fu et al. [42] employed ANN to select efficient updates to accelerate Monte Carlo simulations of classical Ising spin models near critical parameters associated with the phase transition. Similarly, Botu et al. [7] employed kernel ridge regression to accelerate MD method for nuclei-electron systems by learning the selection of probable configurations in MD simulations, which enabled bypassing explicit simulations for several steps. More recently, ML has been used to "short-circuit" calculations based on *ab initio* MD simulations in the area of molecular quantum chemistry [22]. However, relative to "hard" condensed matter systems, a survey of literature finds far fewer applications of ML in the area of MD simulations of soft materials [16, 37].

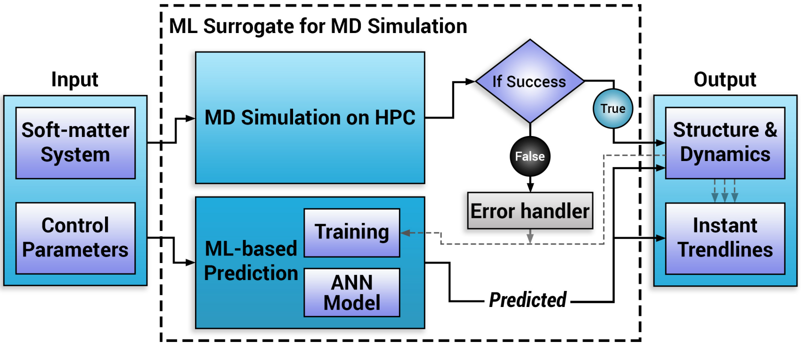


Figure 1: System overview of the ML surrogate for MD simulation approach for generating rapid and accurate predictions associated with soft material behavior.

2.3. ML-enhanced simulations as web applications

nanoHUB is the largest online resource for education and research in nanotechnology [39]. This cyberinfrastructure hosts over 500 web applications for launching simulations and serves 1.4 million users worldwide. nanoHUB provides online access for executing simulation codes to researchers, students, and educators. We have deployed MD simulation frameworks exploring diverse self-assembly phenomena in materials as computational tools on nanoHUB. These are: Ions in Nanoconfinement [38], Nanosphere Electrostatics Lab [35], Nanoparticle Assembly Lab [8], and Polyvalent Nanoparticle Binding Simulator. The “Ions in nanoconfinement” tool [38] has been extensively employed by students to learn nanoscale self-assembly concepts [24]. In about 2 years since its launch, this nanoHUB tool has been used by over 100 users and run over 2600 times [38].

The integration of ML for performance enhancement of scientific simulation frameworks deployed as web applications is relatively far less explored. Our survey indicated that only one simulation tool on nanoHUB [21] employs ML-based methods to enhance the performance and usability of the simulation software. This simulation tool employs a deep neural network to bypass computational limitations in extracting transfer times associated with the excitation energy transport in light-harvesting systems [21].

3. ML Surrogates for MD Simulations

We now describe a general approach, first introduced in Ref. [36], that utilizes ML to enhance MD simulations, significantly improving their use in research and education. The “ML surrogates for MD simulations” framework can be broadly defined as the approach to use ML to learn from MD simulations, and produce learned surrogates for MD simulations. Figure 1 shows the overview of this framework in the context of soft materials engineering applications to predict the structural and dynamical properties (outputs) of the the soft-matter system over a broad range of experimental control parameters (inputs). First, the attributes of the soft-matter system and the control parameters are fed to the framework (Figure 1). These inputs are used to launch the MD simulation on the HPC cluster. Simultaneously, these inputs are fed to the ML-based prediction module. Both the MD and ML methods are designed to extract (predict) the desired output quantities. Error handler aborts the MD simulation program and displays appropriate error messages when a simulation fails due to any pre-defined criteria. At the end of the simulation run, the output quantities are saved for future retraining of the ML model, which occurs after a set number of new successful simulation runs. After yielding a sufficiently large set of predictions, the ML surrogate rapidly provides trendlines capturing the behavior of output quantities as a function of variation in input parameters.

ML surrogates for MD simulations enable several capabilities: (i) learn pre-identified interesting features associated with the simulation outputs, (ii) generate accurate predictions for unseen state points, (iii) enable instantaneous predictions, and (iv) improve interactivity with anytime access to simulation results.

We now describe the application of this framework to the specific case of MD simulations of ions in nanoconfinement to illustrate the approach in further detail. Here, the goal is to extract the distribution of ions confined by two material surfaces represented as identical, uncharged parallel plates at $z = -h/2$ and $z = h/2$ (creating a confinement length of h). The inputs include the attributes of the electrolyte ions such as valency and size, and the control parameters such as electrolyte concentration and interface separation. The outputs include the density profiles or distribution functions of ions in confinement. These parameter choices enable a rich competition between electrostatic and steric forces characterizing the ionic dynamics and structure in nanoconfinement.

Our earlier work [36] showed that an artificial neural network (ANN) model can accurately predict 3 key output features: contact density, peak density, and mid-point density of confined ions. ANN outperformed other ML techniques such as polynomial regression, support vector regression, decision tree regression, and random forest regression. In this paper, we show that the ANN model can generate predictions for nearly all the desired features of the ionic density profile, producing almost the entire ionic distribution in excellent agreement with explicit MD simulation results.

3.1. Data Generation, Preparation, and Preprocessing

Prior domain experience and backward elimination using the adjusted R squared is used for selecting the most significant input parameters for creating the training data set. This process determines which inputs are important to change the desired output. The process begins with all possible inputs and eliminates them one by one, monitoring the adjusted R squared value to determine which are important. Dropping of an input that produces a sharp spike in the R squared value is indicative of the high importance of the input parameter. Using this process, five input parameters characterizing the ionic system are identified: confinement length h, salt (electrolyte) concentration c, positive ion valency z_p, negative ion valency z_n, and the ion diameter d. All ions are assumed to have the same diameter; in general, oppositely charged ions have different sizes. The range of each parameter is selected as follows: $h \in (3.0, 4.0)$ nm, $c \in (0.3, 0.9)$ M, $z_p \in 1, 2, 3$, $z_n \in -1$, and $d \in (0.5, 0.75)$ nm. The salt concentration is defined as the number of negative ions per unit volume [36, 37, 33]. Min-max normalization filter is applied to normalize the input data at the preprocessing stage.

The converged distribution for positive ions is selected as the output. The dataset created for investigations in the earlier work [36] is reused. This dataset was generated by sweeping over a few discrete values for each of the input and output parameters to create and run 6,864 MD simulations utilizing HPC resources. On average, each MD simulation was performed for over ≈ 5 nanoseconds of ionic dynamics, and took 4200 CPU hours (≈ 36 minutes per simulation with MPI/OpenMP parallelization). The training dataset creation took approximately 25 days including the queue wait times on the Indiana University BigRed2 supercomputing cluster. The data associated with the ionic density profiles (dataset relevant to our investigation), was over 2 GB.

The entire data set is separated into training and testing sets using a ratio of 0.8:0.2. In our earlier work, contact density ρ_c, mid-point (center of the slit) density ρ_m, and peak density ρ_p associated with the final (converged) distribution for positive ions were selected as the output parameters [36]. Each MD simulation produces positive ion distribution characterized by ionic density measured at $\approx$ 300 positions as output. For simplicity, using the symmetry of ionic density around the confinement center $z = 0$ (resulting from neutral surfaces), approximately half of the 300 points are selected as the output parameters to train the ML surrogate. Accordingly, in the experiments that follow, ML is employed to make $P \approx 150$ predictions characterizing the density of ions in the left half of the confinement (with $z \in (-h/2, 0)$).

3.2. Feature Extraction and Regression

Following earlier paper [36], the ANN architecture with 2 hidden layers (Figure 2) is implemented in TensorFlow [1] for regression and prediction of $P \approx 150$ continuous (output) variables. The process of regression first determines weights and biases in the two hidden layers following an error backpropagation algorithm, implemented via a stochastic gradient descent procedure. This process employs an iterative learning algorithm that uses a training dataset to update the weights and biases in the hidden layers. Regression is done by a simple forward prediction.

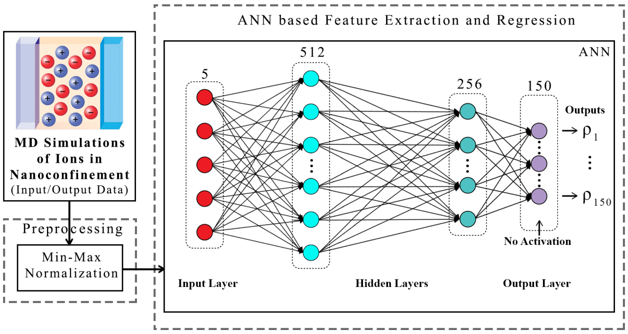


Figure 2: Artificial neural network based regression model used in the ML surrogate to predict output ionic density profile of ions in nanoconfinement.

The size of the hidden layers was chosen to be over 150 informed by the higher dimensions of the output data. By performing a grid search, hyper-parameters such as the number of first hidden layer units, second hidden layer units, batch size, and the number of epochs are optimized to 512, 256, 25, and 4000 respectively. The batch size is a hyperparameter of stochastic gradient descent algorithm that controls the number of training samples that are allowed to pass through the model before its internal parameters are updated. The number of epochs is a hyperparameter that controls the number of complete passes made through the entire training dataset. Adam optimizer is used to optimize the error backpropagation. The learning rate of Adam optimizer and the dropout rate in the dropout layer is set to 0.0001 and 0.15 respectively to prevent overfitting. Both learning and dropout rates were selected using a trial-and-error process.

The weights in the hidden layers and in the output layer are initialized for better convergence using a Xavier normal distribution at the beginning. Xavier normal distribution is characterized with 0 mean and $\sigma = 1/(\sqrt{h_i + h_o})$ variance, where h_i and h_o are input and output sizes of the hidden layers, respectively [17]. The L2 error (mean square loss) between target and predicted ionic density values is used for error calculation. ANN implementation, training, and testing are programmed using scikit-learn, Keras, and TensorFlow ML libraries [14, 11, 1]. Scikit-learn is used for grid search and scaling, Keras is used to save and load models, and TensorFlow is used to create and train the neural network.

4. Results

4.1. Preliminary Results

In our earlier work [36] we experimented with 6 regression models to predict the key output density features identified above: contact density (ρ_c), mid-point density (ρ_m), and peak density (ρ_p). These models were tested on 2060 sets of input parameters (h, z_p, z_n, c, d). Table 1 shows the success

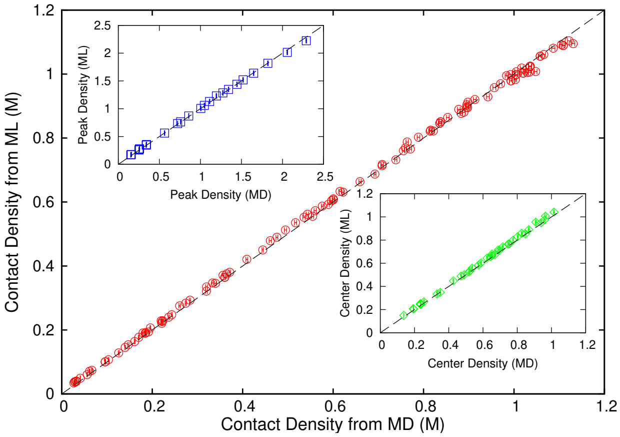


Figure 3: Accuracy comparison between ML predictions and MD simulation results for the contact densities (red circles) of ions in systems characterized by inputs selected from the following ranges of parameters: $h \in (3.0, 4.0)$ nm, $z_p \in 1, 2, 3$, $z_n \in -1, -2$, $c \in (0.3, 0.9)$ M, and $d \in (0.5, 0.75)$ nm. Top-left and bottom-right insets show the comparison for the peak (blue squares) and mid-point (green diamonds) densities respectively for a subset of the selected systems. Black dashed lines with a slope of 1 represent perfect correlation. All densities are shown in units of molars.

rate and the mean square error (MSE) for testing data sets. The success rate was calculated based on the error bars associated with the density values obtained via MD simulations: ML prediction was considered successful when the predicted density value was within the error bar of the simulation estimate. Simulations were run for sufficiently long times (over ≈ 5 nanoseconds) to obtain converged density estimates and error bars. MSE values are calculated using k-fold cross-validation techniques with $k = 20$. k-fold cross-validation is a statistical technique commonly used in applied ML to estimate the accuracy of the ML models. This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining $k - 1$ folds. The test accuracy is determined by taking the average over $(k - 1)$ folds.

ANN-based regression model predicted ρ_c, ρ_m and ρ_p accurately with a success rate of 95.52%, 92.07%, and 94.78% respectively. MSE values for each of these predictions were small as shown in Table 1. ANN outperformed all other non-linear regression models (Table 1) [36].

Figure 3 shows the comparison between the predictions made by the ML model and the results obtained from MD simulations for the contact, mid-point, and peak densities associated with positive ions [36]. For clarity, results are shown for a randomly selected subset of the entire testing dataset described in Section 3.1. ρ_c, ρ_m and ρ_p predicted by the ML model were found to be in excellent agreement with those calculated using the MD method; data from either approach fall on the dashed lines which indicate perfect correlation.

4.2. ML surrogate accuracy and validation

We now describe the extension of the ANN model to make accurate predictions over a much larger set of output ionic density values. In order to demonstrate this, we examine both the overfitting characteristics and the accuracy of the model. Overfitting characteristics are assessed by comparing two losses: training loss on training data and validation loss on test data. Training loss is the average of the MSE between the prediction and the target values for all training data. Validation loss is the average of the MSE between the prediction and the target values for all testing data. The training and validation loss on 5491 and 1373 simulations decrease to 0.000194 and 0.000024 respectively within 4000 epochs of training. Similar amounts of reduction in training and validation losses indicates that the ML model is not overfitted [12].

Typically, in regression problems, accuracy or success rate is inherently not defined. To facilitate the comparison of ML predictions with MD results, we define a prediction as successful when the density value predicted by the ML surrogate ρ^{ML} is within the error ϵ associated with the corresponding MD simulation estimate ρ^{MD} (ground truth). For each simulation, the MD result for the output ionic density is represented by a set of $\approx$ 150 points associated with the discretized positions characterizing the confinement length. The n^{th} prediction made by the ML model corresponds to the n^{th} element in this output set. We define the average success rate or accuracy associated with the ML approach for the n^{th} prediction as:

$$A_n = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \Theta\left(\left|\rho^{\text{ML}}_{n,i} - \rho^{\text{MD}}_{n,i}\right|, \epsilon_{n,i}\right) \qquad (1)$$

where i indicates the simulation index, N_{test} is the number of samples (simulations) in the test data, and $\Theta(x, \epsilon)$ is a step function given by: $\Theta(x, \epsilon) = 1$ for $x < \epsilon$, and $\Theta(x, \epsilon) = 0$ for $x \geq \epsilon$. The overall ensemble-average success rate A of the ML prediction for the entire density profile can be estimated using $A = (1/P) \sum_{n=1}^{P} A_n$, where the sum is now over the total number of predictions P.

Figure 4 shows the success rate A_n associated with the n^{th} ML prediction for the testing dataset. In order for the prediction to be well-evaluated, A_n is only computed for ML predictions associated with non-vanishing ρ^{MD} ($\rho^{\text{MD}} \neq 0$). In other words, results are not shown for $\approx$ 20 output parameters ($P \approx 130$) where the ionic density and associated error from MD simulations are exactly 0 (regions near the left wall where finite-sized ions are prohibited from entering). As Figure 4 shows, A_n is very good for all the evaluated ML predictions. The ensemble-average success rate is found to be $A = 0.958$, and the lowest and highest recorded values for accuracy are $A_n = 0.86$ and $A_n = 0.997$ respectively.

4.3. Ionic density profiles: comparing ML surrogate predictions and MD simulations

We now present plots of the positive ion density profiles showing the comparison between the predictions made by

Table 1
Comparison of regression models for the prediction of output ionic density values.

Model	Contact Density		Midpoint Density		Peak Density	
	Success %	*MSE*	*Success %*	*MSE*	*Success %*	*MSE*
Polynomial	61.04	0.0129300	60.84	0.0187700	61.87	0.0100400
Kernel-Ridge	78.86	0.0030900	76.57	0.0041200	75.93	0.0049800
Support Vector	80.11	0.0012700	79.55	0.0024900	81.98	0.0010600
Decision Tree	68.44	0.0084600	64.54	0.0094900	62.47	0.0110700
Random Forest	74.15	0.0045700	70.85	0.0078900	75.09	0.0040800
ANN-based	95.52	0.0000718	92.07	0.0002293	94.78	0.0002306

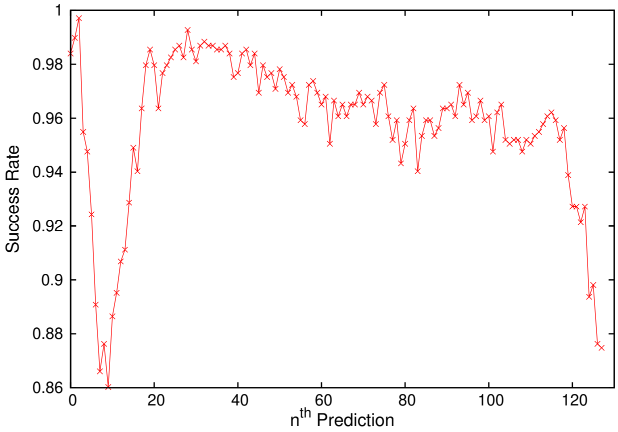


Figure 4: Success rate A_n associated with the n^{th} prediction made the ML surrogate (A_n is defined in Eq. 1).

the ML surrogate and the results obtained from MD simulations. Results are shown for a set of 4 systems randomly selected from the entire testing dataset. These 4 systems are: system I (3.2, 1, -1, 0.6, 0.65), system II (3.6, 3, -1, 0.9, 0.75), system III (3.3, 3, -1, 0.35, 0.714), and system IV (3.6, 1, -1, 0.9, 0.6), where the parentheses list the 5 aforementioned input parameters characterizing the ionic system: confinement length h, positive ion valency z_p, negative ion valency z_n, salt concentration c, and ion diameter d. Figure 5 (a) - (d) shows the ionic density profiles predicted by the ML surrogate for systems I, II, III, and IV respectively. As the figure indicates, for each system, the ML-predicted density profile is in excellent agreement with the result extracted using MD simulation (ground truth).

To make the comparison between ML predictions and MD simulation results more quantitative, we extract the overall accuracy of the ML prediction for each system (density profile). Similar to Eq. 1, we define this accuracy or success rate A_i associated with the i^{th} system (simulation configuration) as:

$$A_i = \frac{1}{P}\sum_{n=1}^{P}\Theta\left(\left|\rho^{\mathrm{ML}}_{n,i} - \rho^{\mathrm{MD}}_{n,i}\right|, \epsilon_{n,i}\right) \qquad (2)$$

where Θ is the step function defined above, and the sum is over the total number of predictions P made using the ML approach (as before, A_i is meaningful only at non-zero ionic density values, making $P \approx 130$). Using Eq. 2, the success rates A_i for systems I, II, III, and IV are found to be 0.98, 0.91, 0.78, 0.89 respectively. In addition to the good success rates, we point out that the ML inferences are made at a relatively much smaller time of ≈ 0.2 seconds compared to MD simulations (see below for further details).

4.4. Instantaneous trendlines using ML surrogates

The good agreement between ionic densities generated via ML surrogate and MD simulations as well as the much smaller "lookup time" for obtaining the ML inferences enable the generation of trendlines for the entire density profile almost instantaneously. Figure 6 shows a selected subset of these ML-surrogate-predicted trendlines exhibiting the variation of ion distributions with changes in input parameters.

Figure 6 (a) and (b) shows the variation in the density of positive ions of valency $z_p = 1, 2, 3$ at salt concentration $c = 0.5$ M and 0.9 M respectively (note that we define $c = N_n/V$, where N_n is the number of negative ions and V is the simulated volume). Other input parameters are fixed to $h = 3.0$ nm, $z_n = -1$, and $d = 0.7$ nm. The ML-generated trendlines are able to track distinct variations in density for different ion valencies and salinity conditions. The peak of the ionic density and the number of oscillations increase with c. Further, at a given c, increasing the ion valency leads to the depletion of ions near the left surface (reduced ion density near $z = 0$). Both these observations inferred by the ML surrogate follow the expected behavior in these systems as reported and elucidated in previous work [33].

4.5. ML inference time and overall speedup

In the earlier paper [36], we introduced a simple formula illustrative of the possible gains or speedup S resulting from the use of scientific ML surrogates:

$$S = \frac{t_{sim}}{t_p + t_{tr} \cdot N_{tr}/N_p}, \qquad (3)$$

where t_{sim} is the time to run the MD simulation via the sequential model, t_p is the time it takes for the ML surrogate to make a prediction (or "lookup" time) for one set of inputs, N_p is the number of ML predictions made, N_{tr} is the number of elements in the training dataset, and t_{tr} is the average walltime associated with the MD simulation to create one

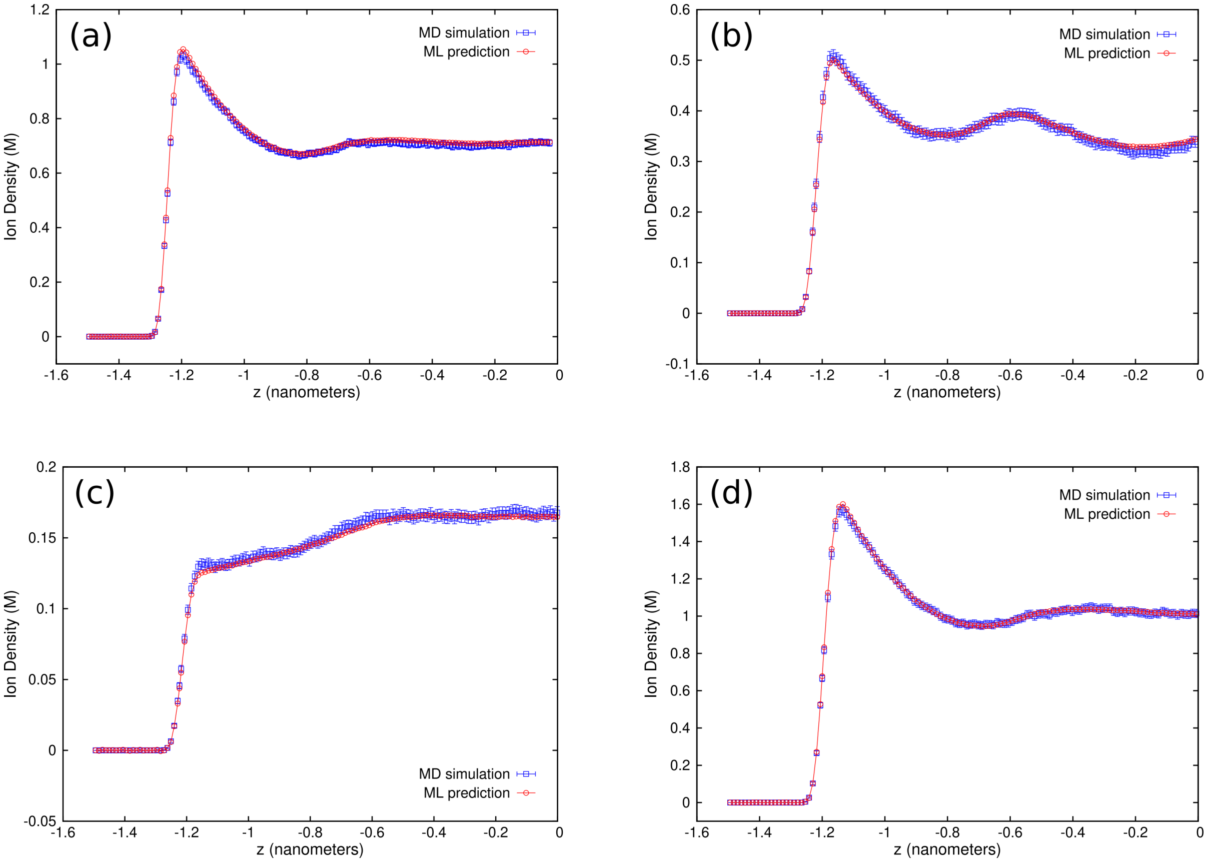


Figure 5: Ionic density profiles for systems I (a), II (b), III (c), and IV (d) predicted by the ML surrogate (red circles) and extracted with MD simulation (blue squares). See main text for system definitions.

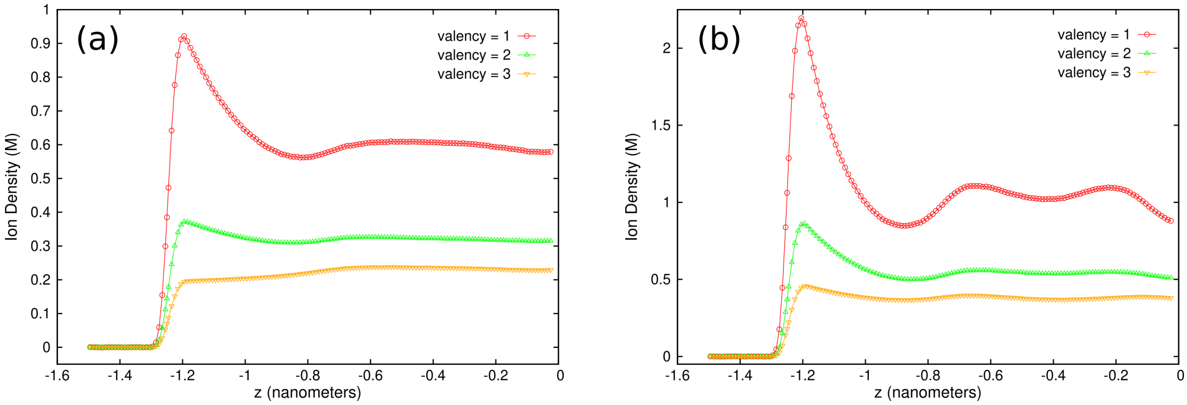


Figure 6: Trendlines generated using ML surrogate to examine variation in ionic density with positive ion valency at salt concentration (a) $c = 0.5$ M and (b) 0.9 M. See main text for values of other input system parameters.

of these elements. $N_{tr}t_{tr}$ is the total time to create the training dataset which includes the generation of the training data using MD simulations and the TensorFlow training time.

In the specific case of the system of confined ions considered above, the training dataset consisted of 5491 simulation configurations ($N_{tr} = 5491$). The time t_{tr} to generate one element of this training set is similar to the average runtime of the parallelized MD simulation. For the MD simulations considered in this work, $t_{sim} \approx 60$ hours, $t_{tr} \approx 36$ minutes, and ML inference time $t_p \approx 0.2$ seconds. Comparing the run times of the parallelized MD simulation and the ML surrogate, we find that the ML surrogate yields output results over 10,000 ($= t_{tr}/t_p$) times faster than parallel MD simulation. We also note that the ML surrogate is using 1 core to infer the result as compared to 128 cores employed by the parallel simulation, indicating a complementary reduction in resource utilization by a factor of 128.

Given $t_p \ll t_{tr}$ and approximating $t_p \rightarrow 0$ in Eq. 3, we find $S = S_{\text{HPC}} N_p / N_{tr}$, where $S_{\text{HPC}} = t_{sim}/t_{tr}$ is the traditional speedup obtained by parallelizing the MD simu-

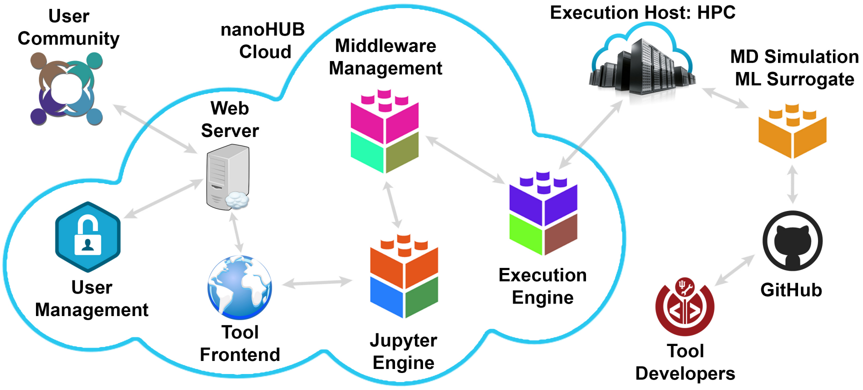


Figure 7: Ecosystem of the computational tool, "Ions in nanoconfinement" [38], deployed on nanoHUB.

lation using HPC resources. We can identify the ML-only speedup as $S_{\text{ML}} = N_p/N_{tr}$, i.e, the number of predictions made by the ML surrogate divided by the size of the training dataset. The key feature of the ML-based approach is thus highlighted: S rises with increasing N_p, that is, the speedup increases as the ML surrogate is used to make more predictions. We also observe that for ML-enabled results to generate a "true" net speedup ($S > 1$), the number of predictions made by the surrogate (ANN model) per one forward propagation must exceed N_{tr}/S_{HPC}, that is, $N_p > N_{tr}/S_{\text{HPC}}$. For the example considered here, $S_{\text{HPC}} = 100$, yielding the relation $N_p \gtrsim 60$. Considering this inequality, the use of ML surrogate enhances the overall efficiency of the simulation framework when it predicts over 60 ionic density profiles.

4.6. ML-enhanced nanoHUB application

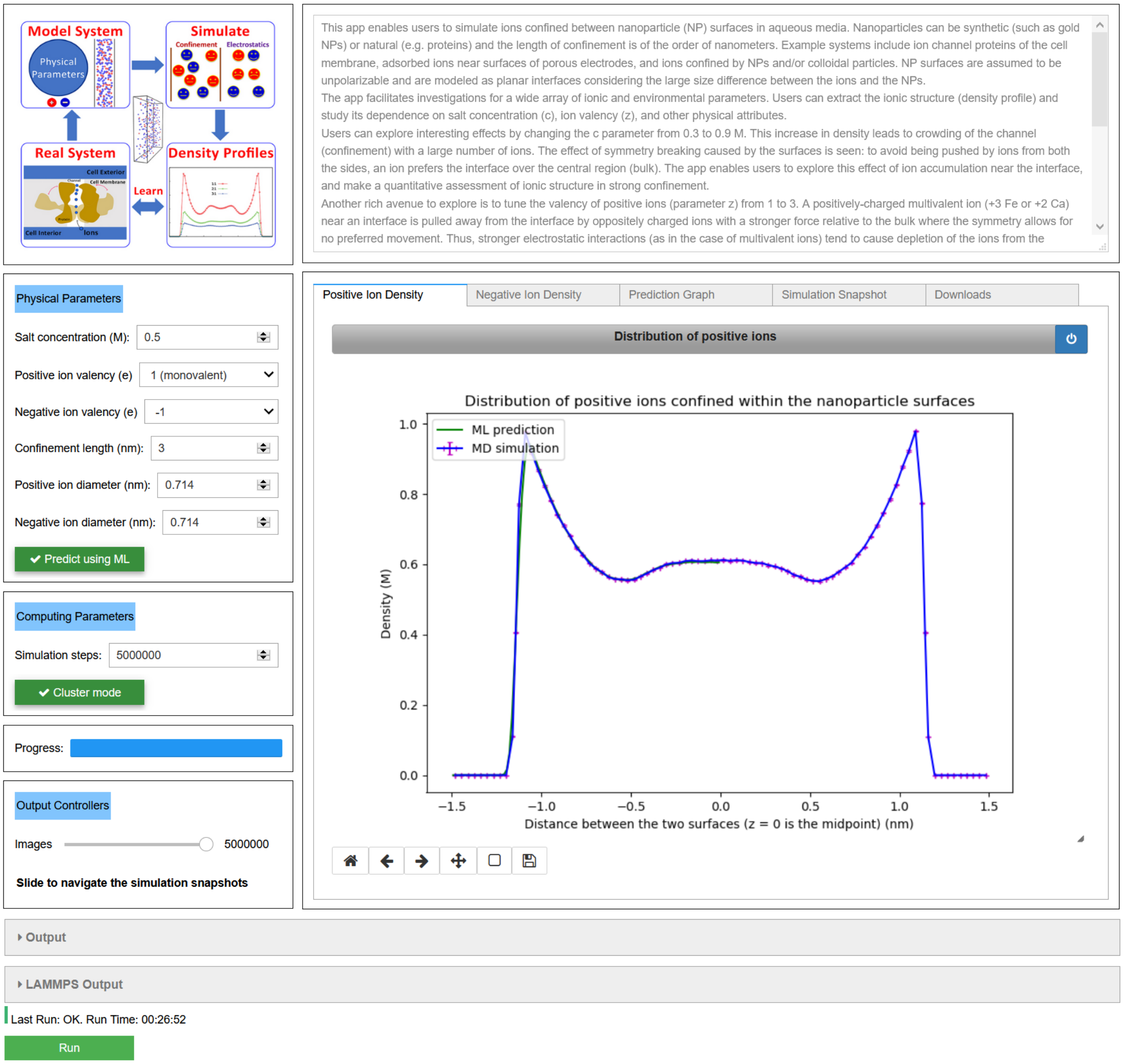
In our original work [34], we proposed to extend the usability of a computational tool, "Ions in nanoconfinement" [38], deployed on nanoHUB (Figure 7) by integrating ML-based enhancements. This tool enables investigations into the self-assembly of coarse-grained models of electrolytes in nanoconfinement, and has been employed to teach courses in Indiana University on nanoscale simulation [24]. Our initial design focused on predicting ionic density for three values/positions (contact, mid-point, peak) [36]. Building on the extension of the ANN model to predict almost all the interesting features of the ionic density profile (with over $\approx$ 150 predicted density values), we designed the integration of the ML surrogate with the nanoHUB tool to predict the left-half of the entire density profile. In October 2019, we deployed this enhanced application using the Jupyter python notebook interface on nanoHUB.

Figure 8 shows the GUI of the deployed tool. Users are provided the choice to click both "Run" and "Predict using ML" buttons simultaneously or separately depending on the desired information. "Predict using ML" activates the ML surrogate which predicts half of the density profile instantaneously. When a user clicks the "Run" button, the Jupyter notebook passes the selected input parameters to the in-house C++ application to do the preprocessing executed on a virtual machine (VM) allocated by middleware management as shown in the Figure 7. Next, the preprocessing script creates the input files required to run the actual MD simulation using the in-house program or LAMMPS [48]. The users are given the option of clicking on "Cluster mode" button for accessing supercomputing resources to lower the computing time. Based on the state of the "Cluster mode" button, execution engine either submits a job for computing clusters or runs the simulation on a VM. When the simulation is over, the execution engine passes the generated data to the postprocessing script to generate the positive and negative ion density profiles. These files are plotted on the "Positive Ion Density" and "Negative Ion Density" tabs, and users are provided the access to download the associated data from the "Downloads" tab.

Users can enable ML surrogate any time by clicking the "Predict using ML" button for accessing the ML-predicted ionic density profile. The predicted profile is displayed in the "Prediction Graph" tab. The same prediction graph is also available as an overlay in the "Positive Ion Density" tab when the MD simulation run is completed. ML surrogate is executed inside the Jupyter notebook environment using TensorFlow ML libraries as a separate thread linked to the "Predict using ML" button. For illustration purposes, Figure 8 shows the final density plot using this integrated MD + ML approach for the input parameters $h = 3.0$ nm, $z_p = 1$, $z_n = -1$, $c = 0.5$ M, and $d = 0.714$ nm.

5. Discussion and Conclusion

In this paper, we explored the idea of using scientific ML surrogates to enhance the usability of MD simulations of soft materials for both research and education. The success of the idea was demonstrated using an example soft-matter simulation framework of self-assembling electrolyte ions in nanoconfinement. The overall success rates and rapid inference times associated with the ML predictions enable an interactive, dynamic, and responsive simulation environment open for wide exploration. To facilitate this possibility, we designed and integrated an ML surrogate with the current version of the "Ions in nanoconfinement" computa-



This app enables users to simulate ions confined between nanoparticle (NP) surfaces in aqueous media. Nanoparticles can be synthetic (such as gold NPs) or natural (e.g. proteins) and the length of confinement is of the order of nanometers. Example systems include ion channel proteins of the cell membrane, adsorbed ions near surfaces of porous electrodes, and ions confined by NPs and/or colloidal particles. NP surfaces are assumed to be unpolarizable and are modeled as planar interfaces considering the large size difference between the ions and the NPs.
The app facilitates investigations for a wide array of ionic and environmental parameters. Users can extract the ionic structure (density profile) and study its dependence on salt concentration (c), ion valency (z), and other physical attributes.
Users can explore interesting effects by changing the c parameter from 0.3 to 0.9 M. This increase in density leads to crowding of the channel (confinement) with a large number of ions. The effect of symmetry breaking caused by the surfaces is seen: to avoid being pushed by ions from both the sides, an ion prefers the interface over the central region (bulk). The app enables users to explore this effect of ion accumulation near the interface, and make a quantitative assessment of ionic structure in strong confinement.
Another rich avenue to explore is to tune the valency of positive ions (parameter z) from 1 to 3. A positively-charged multivalent ion (+3 Fe or +2 Ca) near an interface is pulled away from the interface by oppositely charged ions with a stronger force relative to the bulk where the symmetry allows for no preferred movement. Thus, stronger electrostatic interactions (as in the case of multivalent ions) tend to cause depletion of the ions from the

Physical Parameters
Salt concentration (M): 0.5
Positive ion valency (e): 1 (monovalent)
Negative ion valency (e): -1
Confinement length (nm): 3
Positive ion diameter (nm): 0.714
Negative ion diameter (nm): 0.714

Computing Parameters
Simulation steps: 5000000

Output Controllers
Images 5000000
Slide to navigate the simulation snapshots

Last Run: OK. Run Time: 00:26:52

Figure 8: GUI of the ML-surrogate-enhanced "Ions in nanoconfinement" computational tool deployed on nanoHUB [38]. The GUI includes the density profile predicted by the ML surrogate for half of the position values (in green) for an example ionic system. These results also appear in Prediction graph tab on the GUI.

tional tool deployed on nanoHUB [38].[1] The ML-enhanced nanoHUB tool was used to teach nanoscale simulation concepts in an Indiana University course in Fall 2019. As the "Ions in nanoconfinement" framework evolves with more input features (e.g., surface charge density, asymmetric ion sizes) as well as output quantities (e.g., screening factor), we plan to retrain the ML surrogate to enable predictions for those new input features.

Results from this investigation are encouraging and we intend to explore the feasibility of these ideas in other soft-matter systems. As an example, we plan to explore the design and utility of ML surrogates for MD-based simulated annealing methods used to extract equilibrium shapes of soft-matter-based, deformable nanocontainers [9, 32]. Future work will also explore the extent to which the ML surrogates can predict the desired simulation outputs outside the pre-defined range of training datasets. We also plan to design smart sampling approaches to reduce the amount of training data needed to achieve the desired accuracy associated with the ML surrogate. Finally, we plan to investigate the possibility of extending the idea of an ML surrogate in predicting final outputs to Monte Carlo simulations of materials [42, 26, 25]. We expect that the usefulness of the ML-enabled enhancements demonstrated in this work strengthen the case for scientific simulation applications to be designed and developed with an ML surrogate that learns from the simulations and optimizes the application execution.

[1] The ML-enhanced tool is online since October 2019.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation through Awards 1720625 and DMR-1753182. Simulations were performed using the Big Red II supercomputing system supported in part by Lilly Endowment, Inc., through its support for the IU Pervasive Technology Institute, and in part by the Indiana METACyt Initiative.

Biography

JCS Kadupitiya was born in Sri Lanka in 1990 and completed his Bachelor's degree in Electrical and Information Engineering from the University of Ruhuna, Sri Lanka, in 2015. He received his Masters in Computer Science and Engineering at the University of Moratuwa, Sri Lanka, in 2017. He also received his second Masters in Computer Engineering from Indiana University Bloomington in 2019, and he is currently enrolled in Indiana University Bloomington as a Ph.D. student in the Department of Intelligent Systems Engineering. His research interests lie primarily in scientific machine learning and distributed computing.

Geoffrey Charles Fox received a Ph.D. in Theoretical Physics from Cambridge University where he was Senior Wrangler. He is now a distinguished professor of Engineering, Computing, and Physics at Indiana University. He previously held positions at Caltech, Syracuse University, and Florida State University after being a postdoc at the Institute for Advanced Study at Princeton, Lawrence Berkeley Laboratory, and Peterhouse College Cambridge. He has supervised the Ph.D. of 73 students with a h-index of 78 and over 36000 citations. He is a Fellow of APS (Physics) and ACM (Computing) and works on the interdisciplinary interface between computing and applications.

Vikram Jadhao is an assistant professor of Intelligent Systems Engineering at Indiana University in Bloomington. Prior to joining Indiana University, he held postdoctoral fellowships in the Department of Physics and Astronomy at Johns Hopkins University as well as the Department of Materials Science and Engineering at Northwestern University. He received his Ph.D. and M.S. in Physics from the University of Illinois at Urbana-Champaign, and his B.S. in Physics from the Indian Institute of Technology at Kharagpur. His current research interests are in nanoscale self-assembly, soft-matter simulation, biomimetic materials, and machine learning. He is a recipient of an NSF CAREER award.

References

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al., 2016. Tensorflow: a system for large-scale machine learning., in: OSDI, pp. 265–283.

[2] Abruña, H.D., Kiya, Y., Henderson, J.C., 2008. Batteries and electrochemical capacitors. Phys. Today 61, 43–47.

[3] Allen, R., Hansen, J.P., Melchionna, S., 2001. Electrostatic potential inside ionic solutions confined by dielectrics: a variational approach. Phys. Chem. Chem. Phys. 3, 4177–4186.

[4] Barros, K., Sinkovits, D., Luijten, E., 2014. Efficient and accurate simulation of dynamic dielectric objects. The Journal of Chemical Physics 140, 064903. doi:10.1063/1.4863451.

[5] Behler, J., 2016. Perspective: Machine learning potentials for atomistic simulations. The Journal of chemical physics 145, 170901.

[6] Boda, D., Gillespie, D., Nonner, W., Henderson, D., Eisenberg, B., 2004. Computing induced charges in inhomogeneous dielectric media: Application in a monte carlo simulation of complex ionic systems. Phys. Rev. E 69, 046702.

[7] Botu, V., Ramprasad, R., 2015. Adaptive machine learning framework to accelerate ab initio molecular dynamics. International Journal of Quantum Chemistry 115, 1074–1083.

[8] Brunk, N., Kadupitiya, J., Uchida, M., Douglas, T., Jadhao, V., 2019a. Nanoparticle assembly lab. URL: https://nanohub.org/resources/npassemblylab, doi:doi:10.21981/RECV-RD32. nanoHUB.

[9] Brunk, N.E., Jadhao, V., 2019. Computational studies of shape control of charged deformable nanocontainers. Journal of Materials Chemistry B 7, 6370. URL: http://dx.doi.org/10.1039/C9TB01003C, doi:10.1039/C9TB01003C.

[10] Brunk, N.E., Uchida, M., Lee, B., Fukuto, M., Yang, L., Douglas, T., Jadhao, V., 2019b. Linker-mediated assembly of virus-like particles into ordered arrays via electrostatic control. ACS Applied Bio Mat. 2, 2192–2201. doi:10.1021/acsabm.9b00166.

[11] Buitinck, L., et al., 2013. Api design for machine learning software: experiences from the scikit-learn project. arXiv:1309.0238 .

[12] Cawley, G.C., Talbot, N.L., 2010. On over-fitting in model selection and subsequent selection bias in performance evaluation. Journal of Machine Learning Research 11, 2079–2107.

[13] Ch'ng, K., Carrasquilla, J., Melko, R.G., Khatami, E., 2017. Machine learning phases of strongly correlated fermions. Phys. Rev. X 7, 031038. doi:10.1103/PhysRevX.7.031038.

[14] Chollet, F., et al., 2015. Keras.

[15] Fahrenberger, F., Xu, Z., Holm, C., 2014. Simulation of electric double layers around charged colloids in aqueous solution of variable permittivity. The Journal of Chemical Physics 141, 064902. doi:10.1063/1.4892413.

[16] Ferguson, A.L., 2017. Machine learning and data science in soft materials engineering. Journal of Physics: Condensed Matter 30, 043002.

[17] Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249–256.

[18] Glotzer, S.C., 2015. Assembly engineering: Materials design for the 21st century (2013 pv danckwerts lecture). Chemical Engineering Science 121, 3–9.

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

[20] Hadden, J.A., Perilla, J.R., Schlicksup, C.J., Venkatakrishnan, B., Zlotnick, A., Schulten, K., 2018. All-atom molecular dynamics of the hbv capsid reveals insights into biological function and cryo-em resolution limits. Elife 7, e32478.

[21] Häse, F., Kreisbeck, C., Aspuru-Guzik, A., 2017. Machine learning for quantum dynamics: deep learning of excitation energy transfer properties. Chemical science 8, 8419–8426.

[22] Häse, F., Fdez. Galván, I., Aspuru-Guzik, A., Lindh, R., Vacher, M., 2019. How machine learning can assist the interpretation of ab initio molecular dynamics simulations and conceptual understanding of chemistry. Chem. Sci. 10, 2298–2307. URL: http://dx.doi.org/10.1039/C8SC04516J, doi:10.1039/C8SC04516J.

[23] Jadhao, V., 2018. Self-assembly in nanoconfinement. URL: https://nanohub.org/resources/29671. nanoHUB.

[24] Jadhao, V., 2019. Nanoscale simulations and engineering applications: Applications - self-assembly in nanoconfinement. URL: https://nanohub.org/resources/29671.

[25] Jadhao, V., Makri, N., 2010a. Iterative monte carlo path integral with optimal grids from whole-necklace sampling. The Journal of chemical physics 133, 114105.

[26] Jadhao, V., Makri, N., 2010b. Iterative monte carlo with bead-adapted sampling for complex-time correlation functions. The Journal of chemical physics 132, 104110.

[27] Jadhao, V., Robbins, M.O., 2017. Probing large viscosities in glass-formers with nonequilibrium simulations. Proceedings of the National Academy of Sciences 114, 7952–7957. URL: https://www.pnas.org/content/114/30/7952, doi:10.1073/pnas.1705978114.

[28] Jadhao, V., Robbins, M.O., 2019. Rheological properties of liquids under conditions of elastohydrodynamic lubrication. Tribology Letters 67, 66. URL: https://doi.org/10.1007/s11249-019-1178-3, doi:10.1007/s11249-019-1178-3.

[29] Jadhao, V., Solis, F.J., Olvera de la Cruz, M., 2012. Simulation of charged systems in heterogeneous dielectric media via a true energy functional. Physical review letters 109, 223905.

[30] Jadhao, V., Solis, F.J., Olvera de la Cruz, M., 2013a. Free-energy

functionals of the electrostatic potential for poisson-boltzmann theory. Physical Review E 88, 022305.

[31] Jadhao, V., Solis, F.J., Olvera de la Cruz, M., 2013b. A variational formulation of electrostatics in a medium with spatially varying dielectric permittivity. The Journal of chemical physics 138, 054119.

[32] Jadhao, V., Thomas, C.K., Olvera de la Cruz, M., 2014. Electrostatics-driven shape transitions in soft shells. Proceedings of the National Academy of Sciences 111, 12673–12678.

[33] Jing, Y., Jadhao, V., Zwanikken, J.W., Olvera de la Cruz, M., 2015. Ionic structure in liquids confined by dielectric interfaces. The Journal of chemical physics 143, 194508.

[34] Kadupitige, J.C., Fox, G., Jadhao, V., 2019. Machine learning for auto-tuning of simulation parameters in car-parrinello molecular dynamics, in: APS Meeting Abstracts, p. .

[35] Kadupitiya, J., Brunk, N., Ali, S., Fox, G.C., Jadhao, V., 2018. Nanosphere electrostatics lab. URL: https://nanohub.org/resources/nselectrostatic, doi:doi:10.4231/D34X54J88. nanoHUB.

[36] Kadupitiya, J., Fox, G.C., Jadhao, V., 2019. Machine learning for performance enhancement of molecular dynamics simulations, in: International Conference on Computational Science, pp. 116–130. URL: https://link.springer.com/chapter/10.1007/978-3-030-22741-8$_$9.

[37] Kadupitiya, J., Sun, F., Fox, G.C., Jadhao, V., . Machine learning surrogates for molecular dynamics simulations. Journal of Computational Science, Invited Article (In preparation).

[38] Kadupitiya, K., Anousheh, N., Marru, S., Fox, G.C., Jadhao, V., 2017. Ions in nanoconfinement. URL: https://nanohub.org/resources/nanoconfinement, doi:doi:10.21981/D236-ZZ44. nanoHUB.

[39] Klimeck, G., McLennan, M., Brophy, S.P., III, G.B.A., Lundstrom, M.S., 2008. nanohub.org: Advancing education and research in nanotechnology. Computing in Science Engineering 10, 17–23.

[40] Kremer, K., Grest, G.S., 1990. Dynamics of entangled linear polymer melts: A molecular-dynamics simulation. J. Chem. Phys. 92, 5057–5086.

[41] Limbach, H.J., Arnold, A., Mann, B.A., Holm, C., 2006. ESPResSo – an extensible simulation package for research on soft matter systems. Comp. Phys. Comm. 174, 704–727.

[42] Liu, J., Qi, Y., Meng, Z.Y., Fu, L., 2017. Self-learning monte carlo method. Phys. Rev. B 95, 041101.

[43] Long, A.W., Zhang, J., Granick, S., Ferguson, A.L., 2015. Machine learning assembly landscapes from particle tracking data. Soft Matter 11, 8141–8153.

[44] Luo, G., Malkova, S., Yoon, J., Schultz, D.G., Lin, B., Meron, M., Benjamin, I., Vanýsek, P., Schlossman, M.L., 2006. Ion distributions near a liquid-liquid interface. Science 311, 216–218.

[45] Morningstar, A., Melko, R.G., 2017. Deep learning the ising model near criticality. arXiv preprint arXiv:1708.04622 .

[46] Nygård, K., Sarman, S., Kjellander, R., 2013. Local order variations in confined hard-sphere fluids. The Journal of Chemical Physics 139, –.

[47] Perilla, J., Hadden, J., Goh, B., Mayne, C., Schulten, K., 2016. All-atom molecular dynamics of virus capsids as drug targets. The journal of physical chemistry letters 7, 1836–1844. URL: https://pubs.acs.org/doi/10.1021/acs.jpclett.6b00517.

[48] Plimpton, S., 1995. Fast parallel algorithms for short-range molecular dynamics. Journal of Computational Physics 117, 1 – 19.

[49] Schoenholz, S.S., 2018. Combining machine learning and physics to understand glassy systems. Journal of Physics: Conference Series 1036, 012021.

[50] Smith, A.M., Lee, A.A., Perkin, S., 2016. The electrostatic screening length in concentrated electrolytes increases with concentration. The journal of physical chemistry letters 7, 2157–2163.

[51] Solis, F.J., Jadhao, V., Olvera de la Cruz, M., 2013. Generating true minima in constrained variational formulations via modified lagrange multipliers. Physical Review E 88, 053306.

[52] Spellings, M., Glotzer, S.C., 2018. Machine learning for crystal identification and discovery. AIChE Journal 64, 2198–2206.

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