Published in final edited form as:

J Chem Theory Comput. 2021 October 12; 17(10): 6214–6224. doi:10.1021/acs.jctc.1c00492.

Machine-Learned Molecular Surface and Its Application to Implicit Solvent Simulations

Haixin Wei,
Departments of Materials Science and Engineering, Molecular Biology and Biochemistry, Chemical and Biomolecular Engineering, and Biomedical Engineering, Graduate Program in Chemical and Materials Physics, University of California, Irvine, California 92697, United States

Zekai Zhao,
Departments of Materials Science and Engineering, Molecular Biology and Biochemistry, Chemical and Biomolecular Engineering, and Biomedical Engineering, Graduate Program in Chemical and Materials Physics, University of California, Irvine, California 92697, United States

Ray Luo
Departments of Materials Science and Engineering, Molecular Biology and Biochemistry, Chemical and Biomolecular Engineering, and Biomedical Engineering, Graduate Program in Chemical and Materials Physics, University of California, Irvine, California 92697, United States

Abstract

Implicit solvent models, such as Poisson–Boltzmann models, play important roles in computational studies of biomolecules. A vital step in almost all implicit solvent models is to determine the solvent–solute interface, and the solvent excluded surface (SES) is the most widely used interface definition in these models. However, classical algorithms used for computing SES are geometry-based, so that they are neither suitable for parallel implementations nor convenient for obtaining surface derivatives. To address the limitations, we explored a machine learning strategy to obtain a level set formulation for the SES. The training process was conducted in three steps, eventually leading to a model with over 95% agreement with the classical SES. Visualization of tested molecular surfaces shows that the machine-learned SES overlaps with the classical SES in almost all situations. Further analyses show that the machine-learned SES is incredibly stable in terms of rotational variation of tested molecules. Our timing analysis

Corresponding Author: **Ray Luo** – *Departments of Materials Science and Engineering, Molecular Biology and Biochemistry, Chemical and Biomolecular Engineering, and Biomedical Engineering, Graduate Program in Chemical and Materials Physics, University of California, Irvine, California 92697, United States;* rluo@uci.edu.

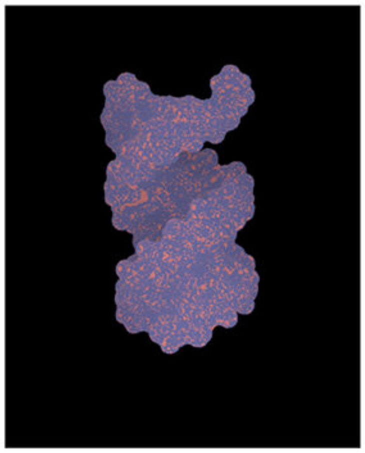
Supporting Information
The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jctc.1c00492.
Training details for the second and the third steps; figures for illustrating the grid labeling scheme in classical SES; incremental training of the ANN model for the second and the third data sets; figures for determining numbers of neurons; superimposed renderings of density function surfaces and classical SES of selected molecules/complexes; renderings of classical SES, machine-learned SES, and density function surfaces of selected molecules/complexes; and rendering of classical SES and machine-learned SES of the GC complex (PDF)
Animation movies for superimposed renderings of machine-learned SES and classical SES of selected molecules/complexes and for superimposed renderings of density function surface and classical SES of selected molecules/complexes (ZIP)

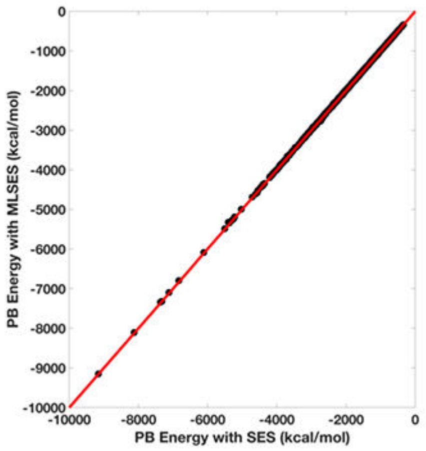
Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jctc.1c00492

The authors declare no competing financial interest.

shows that the machine-learned SES is roughly 2.5 times as efficient as the classical SES routine implemented in Amber/PBSA on a tested central processing unit (CPU) platform. We expect further performance gain on massively parallel platforms such as graphics processing units (GPUs) given the ease in converting the machine-learned SES to a parallel procedure. We also implemented the machine-learned SES into the Amber/PBSA program to study its performance on reaction field energy calculation. The analysis shows that the two sets of reaction field energies are highly consistent with a 1% deviation on average. Given its level set formulation, we expect the machine-learned SES to be applied in molecular simulations that require either surface derivatives or high efficiency on parallel computing platforms.

Graphical Abstract





1. INTRODUCTION

Electrostatic interactions play crucial roles in biophysical processes such as protein and RNA folding, enzyme catalysis, and molecular recognition. Thus, accurate and efficient treatment of electrostatics is vital to computational studies of biomolecular structures, dynamics, and functions. A closely related issue is the modeling of water molecules and their electrostatic interactions with biomolecules that must be considered for any realistic representation of biomolecules under physiological conditions. Implicit solvent model has been such an attempt, in which the solute molecule is treated as a low dielectric constant region with a number of point charges located at atomic centers, and the solvent is treated as a high dielectric constant region. Among all of the attempts, Poisson– Boltzmann (PB) equation-based implicit solvent models have proven to be among the most successful ones and are widely used in computational studies of biomolecules.

A crucial component of all implicit solvent models within the PB framework is the dielectric model, i.e., the dielectric constant distribution of a given solution system. Typically, a solution system is divided into the low dielectric interior and the high dielectric exterior by a molecular surface. That is to say that the molecular surface is used as the dielectric interface between the two piecewise dielectric constants. The solvent excluded surface (SES)[1–3] is the most used surface definition.[4,5] Indeed, previous comparative analyses of PB-based solvent models and TIP3P solvent models show that the SES definition is reasonable in the calculation of reaction field energies and electrostatic potentials of the mean force of hydrogen-bonded and salt-bridged dimers with respect to the TIP3P explicit solvent.[6–8] Given the complexity of the SES, one possible approach in adapting the SES in numerical solutions is to build the molecular surface analytically and then map it onto a grid.[9–11] Since analytical procedures can be time consuming, they are mostly used in the visualization of SES.[3,12–21] Rocchia et al. subsequently converted an analytical algorithm to a semianalytical version with numerical representation of solvent accessible arcs at a very high resolution (i.e., much higher than the finite difference resolution) to facilitate the mapping of the SES to the grid, so the mapping error is negligible in numerical PB calculations.[5] This method is much faster and accurate enough for numerical solution of the PB equation, though it is without an analytical expression.

The van der Waals (VDW) surface, or the hard-sphere surface, represents the low dielectric molecular interior as a union of atomic van der Waals spheres. This is a very efficient algorithm, though there exist many nonphysical high (solvent) dielectric pockets inside the solute interior when the VDW definition is used. Considering the limitation, the modified VDW definition was proposed. The basic idea of the modified VDW definition is to use the solvent accessible surface (SAS) definition for fully buried atoms and the VDW definition for fully exposed atoms.[22] However, the method is difficult to be optimized to reproduce the more physical SES definition.

The density approaches have recently been developed and can be used for numerical PB solutions. Either a Gaussian-like function or a smoothed step function has been explored in previous developments.[23,24] It has been shown that if the functional form is allowed to change, the density function can be explicitly optimized to reproduce the classical SES definition at least for certain “small” solvent probe radii, though this cannot be generalized to arbitrary probe radius values.[25] In these types of approaches, a distance-dependent density/volume exclusion function is used to define each atomic volume or the dielectric constant directly.[26–28] This is in contrast to the hard-sphere definition of atomic volume as in the VDW or the SES definition. Therefore, the surface cusps are removed by the use of smooth density functions.[23,24]

In this study, we intend to address the difficulties of computing SES of complex molecular structures. The difficulties arise because SES is formed by different patches, so that its analytical formulation must be piecewise and localized.[29] There are at least three consequences of the difficulties. First, the algorithms are very costly. For example, the optimized SES still takes about a third of the total central processing unit (CPU) time of the numerical PB solvers in Amber.[30,31] Second, they are hard to be ported to modern parallel platforms, such as graphics processing units (GPUs). Third, it is difficult to obtain surface curvatures or other higher-order surface parameters that are often used for implicit solvent simulations. To overcome the difficulties, our solution is to use a smooth level set function to approximate SES. As will be shown below, the single-thread CPU cost can be reduced noticeably when the level set function is used. Given the level set function, the computation of SES can be carried out for all grid points independently and simultaneously and thus achieve parallel computation. Finally, surface curvatures or other higher-order surface parameters are readily available from the smooth level set function.[32]

Given the complexity of the SES algorithms, we explored a machine learning strategy to look for a smooth level set function. Indeed, in recent years, deep learning techniques have been widely adapted in handling multivariable and highly nonlinear functions similarly challenging such as SES. Deep learning/neural network is inspired by and resembles the human brain. The input variables are multiplied by the respective weights and then undergo a transformation based on an activation function to obtain the outputs.[33] It has been mathematically proven that a single hidden layer was able to solve any continuous problem.[34] With all its advantages, deep learning has been applied in various biological fields, including gene expression,[35–42] protein secondary and tertiary structures,[43–55] protein–protein interactions,[56–62] and others.[63]

In the following, we first overview the necessary physical and mathematical concepts needed to set up a machine learning SES model, along with the computational details in collecting the training data and conducting the training process. This is followed by accuracy analyses of the machine-learned SES, robustness analyses, timing analysis, and finally applications to implicit solvent simulations. We end the presentation with a discussion of potential future developments.

2. METHODS

2.1. Finite Difference Method for Solving PB Equation.

As widely adopted for numerically solving partial differential equations, the finite difference method uses a uniform Cartesian grid to discretize the PB equation. The grid points are numbered as (i,j,k), where $i = 1, \ldots, x_m$, $j = 1, \ldots, y_m$, $k = 1, \ldots, z_m$, and x_m, y_m, and z_m are the numbers of points along the three axes. The grid spacing between neighboring points can be uniformly set to h. To discretize the linearized PB equation

$$\nabla \cdot \varepsilon \nabla \phi - \lambda \sum_i n_i q_i^2 \phi / kT = -4\pi\rho \tag{1}$$

where the charge density ρ can be expressed as $q(i,j,k)/h^3$, $q(i,j,k)$ is the total charge within the cubic volume centered at (i,j,k), and λ is a masking function for the Stern layer. In the salt-related term, n_i is the number density of the ion of type i in the bulk solution, q_i is the charge of the ion of type i, k is the Boltzmann constant, and T is the temperature. The final discretized PB equation can be expressed as follows, if we ignore the salt-related term as it is often modeled to be away from the molecular surface

$$\begin{aligned}
&\left\{\varepsilon\left(i-\frac{1}{2},j,k\right)[\phi(i-1,j,k)-\phi(i,j,k)]\right.\\
&+\varepsilon\left(i+\frac{1}{2},j,k\right)[\phi(i+1,j,k)-\phi(i,j,k)]\\
&+\varepsilon\left(i,j-\frac{1}{2},k\right)[\phi(i,j-1,k)-\phi(i,j,k)]\\
&+\varepsilon\left(i,j+\frac{1}{2},k\right)[\phi(i,j+1,k)-\phi(i,j,k)]\\
&+\varepsilon\left(i,j,k-\frac{1}{2}\right)[\phi(i,j,k-1)-\phi(i,j,k)]\\
&\left.+\varepsilon\left(i,j,k+\frac{1}{2}\right)[\phi(i,j,k+1)-\phi(i,j,k)]\right\}/h^2\\
&=-4\pi q(i,j,k)/h^3
\end{aligned} \tag{2}$$

where $\phi(i,j,k)$ is the potential at grid (i,j,k), and $\varepsilon\left(i-\frac{1}{2},j,k\right)$ is the dielectric constant at the mid-point of the grids (i,j,k) and $(i-1,j,k)$. All other ϕ and ε are defined similarly here.

If a grid point and all its six neighbors are in the same region, either in solvent or in solute, the point is termed a regular grid point. Clearly, the simple discretization scheme above can be applied only to regular grid points because only then the mid-point dielectric constant can be determined (equal to either solvent or solute dielectric constant). However, for an irregular grid point, which means that it has at least one neighboring grid point belonging to the different region, complication arises. Thus, how to define and determine the interface parameters, and then to determine the dielectric constant based on the interface information, is a key for setting up the linearized PB equation. Without question, SES is the most adopted interface definition, as it was found to reasonably reproduce both energetic and dynamic properties of solvated molecules in explicit solvent.[6–8,25] After choosing an interface definition, the discretization can then be handled with many different schemes. For example, the harmonic averaging method is a class of such strategies widely used in biomolecular simulations.[64–66]

2.2. Level Set Functions for Interface Definition.

It is often more convenient to use a level set function to represent an interface.[32] The level set function is often in the form of a second-order continuous distribution function φ, satisfying

$$\begin{aligned}
&\varphi(x,y,z)<0,\ \text{when}(x,y,z)\in\mathbf{\Omega}^-\\
&\varphi(x,y,z)=0,\ \text{when}(x,y,z)\in\mathbf{\Gamma}\\
&\varphi(x,y,z)>0,\ \text{when}(x,y,z)\in\mathbf{\Omega}^+
\end{aligned} \tag{3}$$

Here, $\mathbf{\Gamma}$ represents the interface, and $\mathbf{\Omega}^-$ and $\mathbf{\Omega}^+$ denote the solute and solvent regions, respectively. For example, a signed distance function representing a sphere interface, centered at (x_0,y_0,z_0) with a radius of R, can be expressed as

$$\varphi(x,y,z)=\sqrt{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}-R \tag{4}$$

Level set functions can be used to determine the interface by setting $\boldsymbol{\varphi} = 0$ conveniently but also to compute various interface parameters, such as the normal and tangential directions on the interface as[32]

$$\begin{aligned}\boldsymbol{\xi} &= (\varphi_x, \varphi_y, \varphi_z)\\ \boldsymbol{\eta} &= (\varphi_y, -\varphi_x, 0)\\ \boldsymbol{\tau} &= (\varphi_x\varphi_z, \varphi_y\varphi_z, -\varphi_x^2 - \varphi_y^2)\end{aligned} \tag{5}$$

Here, $\boldsymbol{\xi}$, $\boldsymbol{\eta}$, and $\boldsymbol{\tau}$ are the normal and two tangential directions, respectively. In addition, some advanced discretization schemes, like IIM, [67–70] also require higher-order interface parameters, such as surface curvatures that can also be computed as follows

$$\begin{aligned}\xi_{\eta\eta} &= -\frac{\varphi_{\eta\eta}}{\varphi_\xi}\\ \xi_{\tau\tau} &= -\frac{\varphi_{\tau\tau}}{\varphi_\xi}\\ \xi_{\eta\tau} &= -\frac{\varphi_{\eta\tau}}{\varphi_\xi}\end{aligned} \tag{6}$$

In summary, level set functions are convenient mathematical tools in handling interface-related problems: they can be used to obtain the interface itself and all its geometry parameters in a straightforward manner.[32] Thus, it would greatly benefit discretization of the PB equation if we can express SES as a level set function. However, construction steps of SES are simply too complex to be reproduced algebraically, almost impossible to be done by humans. Therefore, it is better to adopt a machine-learning strategy.

2.3. Deep Learning and Neural Network.

Artificial neural networks (ANNs), usually simply called neural networks, are mathematical models that have been motivated by the brain function.[33,34] A simple example of an ANN structure is shown in Figure 1.

The basic idea of a neuron model is that an input, $\boldsymbol{x}$, together with a bias, b, is weighted by $\boldsymbol{w}$ and then summed together,[71] as shown schematically in Figure 1. The bias, b, is a scalar value, whereas the input $\boldsymbol{x}$ and the weights $\boldsymbol{w}$ are vectors, i.e., $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{w} \in \mathbb{R}^n$ with $n \in \mathbb{N}$ corresponding to the dimension of the input. The sum of these terms, i.e., $z = \boldsymbol{w}^T\boldsymbol{x} + b$, forms the argument of an activation function, $f()$, resulting in the output of the neuron model

$$y = f(z) = f(w^{T}x + b) \tag{7}$$

The role of the activation function, $f()$, is to perform a nonlinear transformation of z. There are many activation functions in practice, such as sigmoid, Tanh, ReLU, Leaky ReLU, and so on, and the ReLU activation function is usually the most popular activation function for deep neural networks.[71]

Though its structure may seem highly complicated, a neural network can be viewed as a combination of simple elementary functions. Thus, a neural network is differentiable to any order, as all its component functions are differentiable to any order, except for a countable number of points. This is an important reason why we have chosen a machine learning approach to express the SES level set function, because it ensures that the SES level set function can be used to compute surface derivatives to any order as needed.

Due to their excellent mathematical properties, ANNs have shown great promise in a range of applications.[63] Most ANN applications fall into two categories: function approximation/regression analysis and classification problems. In this study, the ANN is applied as a classification tool because the goal is to know which region a grid point belongs to (solvent or solute region) in a PB system.

3. COMPUTATIONAL DETAILS

The training data were generated from the Amber PBSA benchmark suite of 573 biomolecular structures.[30] A modified PBSA program was used to print out the training data. The default atomic cavity radii were read from the topology files. The solvent probe radius was set to 1.4 Å, the grid spacing to 0.95 Å, and all other parameters remain as default in the PBSA module in the Amber 20 package.[72] The training software package is Keras in TensorFlow, version 2.2.4.[73] Adam was chosen as the optimizer, and the squared hinge function was chosen as the loss function. The partition ratio of training and validating sets is 9:1, and an early stopping criterion of 100 epochs was used.

The training data contain two parts: labels or the target values, and the variables. The labels are integer values of irregular grid points generated by the Amber/PBSA surface builder for the geometry-based SES, termed classical SES below.[65] This implementation follows the basic idea as outlined by You et al.[10] and Rocchia et al.[5] Specifically, a numerical representation of the solvent accessible arcs was used to map the re-entry surface to the finite difference grid. Briefly, there are two steps in assigning the integer labels. The first is to determine the solvent accessible arcs that are numerically represented centers of solvent accessible probes tangential to at least two atoms simultaneously. The density of the numerical arcs is set to be high enough, so that the mapping error is negligible in numerical PB calculations. The second is to label grid points nearby the molecular surface as inside or outside with a multistep labeling scheme, as summarized in Figure S1.[65]

The irregular points of the outside (solvent) region are labeled as −1 and those of the inside (solute) region are labeled as +1. The variables of the training data are the tuples of coordinates and the radii of those atoms near the irregular grid points in the unit of Ångstrom. An atom is considered "near" a grid point if the distance between the grid point and the atom is within $R + 2D_p$, where R is the atom radius and D_p is the diameter of the probe. The coordinates of the atoms are expressed in the relative frame whose origin is the grid point of interest. The neighbor atoms were then sorted in an ascending order based on the distance between the grid point and their surfaces. In the training data, the maximum neighbor atoms are 48, so the maximum number of variables are 192. The variable dimensions are uniformly enlarged to 200 by filling in zeros for blanks. Overall, there are about 6 million entries of data generated from the Amber PBSA benchmark suite, which was separated into six subsets, each about 1 million entries.

The training was conducted in three steps. First, the model was trained incrementally by adding one atom at a time, which means in the first round of training, the model was fed with only the first (nearest) atom's coordinates and radius. After the training reaches stabilization, the model was fed with the first two atoms' data, and so on. After enough number of atoms were fed to the atom, the accuracy of the model reaches a certain stable level. We then fed it with all of the nearby atoms to complete the training. This step is necessary to initialize the ANN model training because, for such a high-dimension (200) nonlinear system, its local minimum can be extremely deep, causing the training optimization to fail.

In this step, we also investigated how many hidden layers are needed to reach stable prediction accuracy, starting with one layer to five layers. As shown in Figure 2, the accuracy of the model becomes stable at 16 atoms, no matter how many hidden layers were used in the ANN. Therefore, after 16 atoms, all available nearby atoms were fed to the model. Figure 2 further shows that two hidden layers are already very good for our problem. The second and the third steps are to determine the number of neurons (Figure S3) and to finalize the model, respectively. These details can be found in the Supporting Information. The training script and final model can be found at http://rayluolab.org/ml-ses/.

4. RESULTS AND DISCUSSION

4.1. Model Accuracy Analysis.

To evaluate the overall performance of the machine-learned SES method, we first conducted an accuracy test on different structures. The test data sets were generated in the same fashion as the training data sets. Except for the original training protein monomers, we also included two extra sets of biomolecules for this analysis, nucleic acid structures[31] and protein/protein complex structures from previous PBSA developments.[74] The testing results are shown in Table 1.

Table 1 shows that the machine-learned SES performs very well for all structural sets, including those not included in training. The classification accuracies are basically around 96% for all data sets, with a variance of about 1%. The method performs the best for training data set 6 and the nucleic acid data set. This is because those two contain mostly the smallest molecules among all tested molecules. In general, a small molecule has simpler surface geometry, so it is easier to predict its geometries. On the contrary, the protein/protein complex structures are much larger and thus have more complicated surfaces, so the method performs slightly worse on the data set but still obtains an over 95% accuracy.

Another point worth mentioning is that the accuracy test clearly shows excellent transferability of the method, from the smaller training protein monomers to the nucleic acid structures find the larger protein/protein complex structures. The testing molecules overall do not resemble those in the training sets, such is the nucleic acids. Nevertheless, the method can successfully predict the SES of those unseen structures, showing that the method has indeed learned the fundamental rules for predicting the SES surface. In summary, the consistent accuracy confirms that the machine-learned SES method can be applied to typical computational analyses of biomolecules.

4.2. Consistency with Classical SES Molecular Surface.

We chose six representative biomolecules and compared their molecular interfaces generated with three different methods: the newly developed machine-learned SES, the classical geometry-based SES, and a revised density function surface.[25] The superimposed surfaces of the machine-learned SES and the classical SES are shown in Figure 3. The superimposed surfaces of the classical SES and the revised density function are shown in Figure S4 in the Supporting Information. Standalone surfaces from all three methods are also included in Figures S5–S7.

It is clear from Figure 3 that the machine-learned SES excellently reproduces the classical SES for all tested molecules, including both proteins and nucleic acids, consistent with the accuracy analysis shown in Section 4.1. On the contrary, Figure S4 shows that the revised density function does not perform well in reproducing the classical SES for the tested systems. This is because the revised density function was found to agree well with the classical SES only with a smaller solvent probe, i.e., 0.7 Å.[25] At the default solvent probe of 1.4Å, the density surfaces are uniformly “fatter” than SES surfaces. Another limitation in the revised density function surface is that there are often deep re-entry surface patches with very large curvature, which would cause numerical instability when it is used in numerical PB continuum solvents.[70]

4.3. Robustness Analysis.

4.3.1. Rotational Symmetry.—Because SES is a molecular surface, it should possess both translational and rotational symmetries. The translational symmetry is naturally included in the machine-learned SES, as the input coordinates are relative to the grid point and are invariant to the translational transformations. However, the rotational symmetry is in general not preserved if it is not imposed. Thus, if the machine-learned SES can reproduce the classical SES, it should be able to “learn” the rotational symmetry through the training process. Thus, whether the method possesses rotational symmetry is a direct test of whether it has learned the essentials of the classic SES.

To test the rotational symmetry of the machine-learned SES, 20 000 interface points are randomly selected from the training data set and tested with different orientations imposed. Without loss of generality, the *z*-axis was chosen as the rotational axle to generate different orientations. Specifically, we performed the following transformation on the input coordinates of the nearby atoms for each grid point

$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= y\cos(\theta) + x\sin(\theta) \end{aligned}$$

Here, θ is the rotational angle and its values are 30°, 60°, and all the way through 330°. Note that the input coordinates are already transformed with the grid point as the origin (see

Section 3). Next, the trained machine-learned SES is used to predict the grid labels, and the agreement rate with respect to the original predicted grid labels is then computed (Table 2).

As shown above, our machine-learned SES indeed preserves the rotational symmetry because the agreement (aka precision) rate is already higher than the accuracy rate (~95%). As we have discussed above, this is achieved without any human intervention. The analysis shows that the machine-learned SES has learned the essentials of the classic SES very well. In addition, if we look at the actual level set function values that the machine-learned SES gives, for about 95% of the tested grid points, the deviations between the rotated inputs and original inputs are less than 0.001 and for 97% of the tested grid points, the deviations between the two are less than 0.01. Thus, our model not only gives the correct classification but also gives almost the same level set value for different orientations.

4.3.2. Molecular Binding and Unbinding.—

We further evaluated how the machine-learned SES behaves in binding/unbinding scenarios, with a DNA base pair (GC, a pair of guanine and cytosine bases) as a test case. Starting from the hydrogen-bonded GC dimer, the two bases are manually pulled apart by 0.5 Å each step to see how the two SES methods agree with each other for each of the tested conformations.

It is clear from Figure 4 that the machine-learned SES and the classical SES agree excellently for all base separation distances tested, consistent with the performance for single molecules in Section 4.2. For example, both surfaces first generate a small cavity on the right and then another one on the left. The two cavities gradually increase and finally break all three hydrogen bonds between the two bases, causing the two molecules to separate completely. On the other hand, the two surfaces still differ somewhat during the unbinding process. This can be seen more clearly in Figures S8 and S9 that the machine-learned SES splits the complex one step earlier than the classical SES. However, whether these details are of any physical significance is subject to debate. For example, in Figure S8e, the classical SES contains spikes in the middle of the surface, which is obviously not close to physical reality. In contrast, the machine-learned SES does not contain the spikes and as it is produced by a smooth function. The binding/unbinding test shows that the machine-learned SES may be numerically more stable than the classical SES in the more challenging dynamical process.

4.3.3. Extrapolation Analysis.—

To further test the robustness of our model, we want to know how the model performs away from the interface (solvent region or deeply buried molecular interior) since the ANN model was initially trained with irregular points near the solute–solvent interface only (see Section 3).

We have tested the performance of the model in the entire solvation box, not just nearby the interface, but included those grid points that are either far away outside the solute or deeply buried inside the solute. The performance of the model is shown in Figure 5.

Figure 5 shows that the model clearly does not work at grid points far away from the interface. Specifically, Figure 5a illustrates the presence of an “island” or a small cluster of predicted interface grid points, outside the interface. Even more serious failures are within the molecule interior, as illustrated in Figure 5c, where hundreds of small clusters of interface grid points are wrongly predicted. This is surprising because the model can predict with an accuracy level of over 95% in the most difficult interface region yet made hundreds of mistakes in the much easier regions.

These failures are due to the lack of similar data in the training of our initial model as only irregular grid points were fed to the training of the model. Thus, the remedy is to assembly a supplementary training set with grid points not just near the interface, i.e., the irregular grid points in the initial training. In the inside region, the supplementary set includes all grid points that are inside SES but outside VDW. In the outside region, the set includes all grid points that are in the re-entry cones formed by solvent accessible arcs and their associated atom pairs, as illustrated in Figure S1. Training of the initial model on the supplementary set leads to the second version of the model. Without surprise, the second version of the model can successfully classify those grid points incorrectly predicted by the initial model, as shown in Figure 5b,d.

In summary, the extrapolation analysis also demonstrates the robust nature of the machine-learned SES, as it not only works well near the interface but can also be applied to the entire solvation box.

4.4. Timing Analysis.

To illustrate the applications of the machine-learned SES for biomolecular studies, we implemented it into the AMBER/PBSA program. It follows the same algorithm structure as the revised density function.[25] In both approaches, the goal is to assign level set function values for all grid points nearby a molecular solute.[25] The algorithm contains two parts: preprocessing and model predicting. The preprocessing part includes all of the preparation needed for generating inputs for the machine-learned level set function. First, it collects all grid points that fall inside the SAS but outside the VDW of each solute atom. This step involves a loop over all solute atoms. Second, it generates a list of neighboring atoms of each grid point, which were collected in the first step, and then sorts them in the order of distances. The preprocessing part is relatively quick and takes only about 5% of the total time. The predicting part is the main bottleneck of the model, involving a loop over all collected grid points.

Given the current CPU implementation based on the Keras modules, we conducted a timing analysis of the method with the Amber/PBSA benchmark suite used in the training.[30] The classical SES was computed with the default SES builder in the Amber/PBSA program, which is already highly optimized.[65] Figure 6 compares both the timing data of methods. The regression shows that the machine-learned SES method is about 2.5 times as efficient as the classical SES algorithm on the tested compute node on our local cluster.

As discussed in Section 1, an advantage of the machine-learned SES is that it is naturally suitable for parallel computing because the level set function calls for the grid points are independent of each other and thus can be distributed to different computing cores. In other words, the computation cost of the machine-learned model in principle scales almost linearly with the number of cores, so that its cost can be dramatically reduced on highly parallel platforms such as GPUs. Of course, further development to optimize the code and to port it to the GPU platform is necessary to realize its full potential, as observed for the PB solvers in our previous developments.[66,75,76]

4.5. Application to Poisson–Boltzmann Modeling.

The PB reaction field energies were computed with the classical SES, the machine-learned SES, and the revised density function with otherwise identical conditions for all protein structures in the AMBER/PBSA benchmark suite.[30] Figure 7 shows the agreement between the computed PB energies with both machine-learned SES and the revised density function surfaces and those with the classical SES surface.

It is clear from Figure 7 that the machine-learned SES performs uniformly well when used for PB energy calculations. The deviations between the PB energies with the machine-learned SES and those with the classical SES are around 1%. In contrast, the average deviation is roughly 2% but the maximum deviation is over 10% between the PB energies with the density function surface and those with the classical SES. Figure 7c,d further shows that the PB energies from the machine-learned SES are more random around the mean of zero but the PB energies from the density function are systematically more negative. This is because a small solvent probe (0.7 Å) has to be used in the density function to achieve reasonable agreement with the PB energies with the classical SES.[25] If the default probe of 1.4 Å is used, the PB energies would be 30% more positive than those with the classical SES.

5. CONCLUSIONS

In this study, we developed a new level set formulation to generate the solvent excluded surface through a machine learning process. SES is a widely used molecular surface definition, and there are at least two advantages of expressing SES in a level set formalism: to facilitate parallel computing and to make it differentiable for future applications in implicit solvent simulations.

The level set function was trained on the data generated from a set of heterogeneous biomolecular structures, containing about 6 million entries. The training was conducted in three steps to determine the number of nearby atoms needed to define the level set, the number of hidden layers, and the number of neurons of each layer. After the final training, the machine-learned SES can predict the classical SES with an accuracy of over 95% on tested proteins, nucleic acids, and complex structures.

Analysis of visualized molecular surfaces of tested biomolecules shows that the machine-learned SES agrees excellently with the classical SES. It is also clear that a previous approach based on atomic density functions is clearly insufficient, generating a molecular surface often larger than the classical SES and causing deeper crevices in the re-entry regions. Besides, our machine-learned model is incredibly stable in terms of rotational and translational variations of the molecules.

We also performed a timing analysis between the machine-learned SES and the classical SES algorithm. The analysis shows that the machine-learned SES is roughly 2.5 times as efficient as the classical SES routine in AMBER/PBSA on the tested CPU platform. We expect further performance gain on massively parallel platforms such as GPUs given the ease in converting the machine-learned SES to a parallel procedure.

To further test the machine-learned SES, we implemented it into a PB solver to see if it can be used to reproduce the PB reaction field energies computed with the classical SES. Our analysis shows that the two sets of energies differ on average only about 1%, better than the 2% average deviation when the energies between the density function surface and the classical SES are compared. Furthermore, the deviations in PB energies by the machine-learned SES are uniformly distributed, yet the deviations in PB energies by the density function surface often exhibit large deviations as high as 10%. This shows that the machine-learned SES is much more stable in reproducing the classical SES in real-world applications.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

ACKNOWLEDGMENTS

This work was supported by the National Institute of Health/NIGMS (Grant nos. GM093040 and GM130367).

REFERENCES

(1). Richards FM Areas, volumes, packing, and protein structure. Annu. Rev. Biophys. Bioeng 1977, 6, 151–176. [PubMed: 326146]

(2). Connolly ML Solvent-accessible surfaces of proteins and nucleic acids. Science 1983, 221, 709–713. [PubMed: 6879170]

(3). Connolly ML Analytical molecular surface calculation. J. Appl. Crystallogr 1983, 16, 548–558.

(4). Gilson MK; Sharp KA; Honig BH Calculating the electrostatic potential of molecules in solution - method and error assessment. J. Comput. Chem 1988, 9, 327–335.

(5). Rocchia W; Sridharan S; Nicholls A; Alexov E; Chiabrera A; Honig B Rapid grid-based construction of the molecular surface and the use of induced surface charge to calculate reaction field energies: Applications to the molecular systems and geometric objects. J. Comput. Chem 2002, 23, 128–137. [PubMed: 11913378]

(6). Swanson JMJ; Mongan J; McCammon JA Limitations of atom-centered dielectric functions in implicit solvent models. J. Phys. Chem. B 2005, 109, 14769–14772. [PubMed: 16852866]

(7). Tan C; Yang L; Luo R How well does Poisson-Boltzmann implicit solvent agree with explicit solvent? A quantitative analysis. J. Phys. Chem. B 2006, 110, 18680–18687. [PubMed: 16970499]

(8). Wang J; Tan C; Chanco E; Luo R Quantitative analysis of Poisson–Boltzmann implicit solvent in molecular dynamics. Phys. Chem. Chem. Phys 2010, 12, 1194–1202. [PubMed: 20094685]

(9). Zauhar RJ; Morgan RS Computing the electric-potential of biomolecules - application of a new method of molecular-surface triangulation. J. Comput. Chem 1990, 11, 603–622.

(10). You T; Bashford D An analytical algorithm for the rapid determination of the solvent accessibility of points in a three-dimensional lattice around a solute molecule. J. Comput. Chem 1995, 16, 743–757.

(11). Eisenhaber F; Argos P Improved strategy in analytic surface calculation for molecular systems: Handling of singularities and computational efficiency. J. Comput. Chem 1993, 14, 1272–1280.

(12). Varshney A; Brooks FP; Wright WV Computing smooth molecular surfaces. IEEE Computer Graphics and Applications 1994, 14, 19–25.

(13). Edelsbrunner H; Mücke EP Three-dimensional alpha shapes. ACM Trans. Graphics 1994, 13, 43–72.

(14). Totrov M; Abagyan R The contour-buildup algorithm to calculate the analytical molecular surface. J. Struct. Biol 1996, 116, 138–143. [PubMed: 8742735]

(15). Sanner MF; Olson AJ; Spehner JC Reduced surface: An efficient way to compute molecular surfaces. Biopolymers 1996, 38, 305–320. [PubMed: 8906967]

(16). Krone M; Bidmon K; Ertl T Interactive visualization of molecular surface dynamics. IEEE Trans. Visualization Comput. Graphics 2009, 15, 1391–1398.

(17). Lindow N; Baum D; Prohaska S; Hege HC Accelerated visualization of dynamic molecular surfaces. Computer Graphics Forum 2010, 29, 943–952.

(18). Krone M; Grottel S; Ertl T In Parallel Contour-Buildup Algorithm for the Molecular Surface, 2011 IEEE Symposium on Biological Data Visualization (BioVis); IEEE, 2011; pp 17–22.

(19). Parulek J; Viola I In Implicit Representation of Molecular Surfaces, 2012 IEEE Pacific Visualization Symposium; IEEE, 2012; pp 217–224.

(20). Jurčík A; Parulek J; Sochor J; Kozlikova B In Accelerated Visualization of Transparent Molecular Surfaces in Molecular Dynamics, 2016 IEEE Pacific Visualization Symposium (PacificVis); IEEE, 2016; pp 112–119.

(21). Kozlíková B; Krone M; Falk M; Lindow N; Baaden M; Baum D; Viola I; Parulek J; Hege HC Visualization of biomolecular structures: State of the art revisited. Computer Graphics Forum 2017, 36, 178–204.

(22). Lu Q; Luo R A Poisson-Boltzmann dynamics method with nonperiodic boundary condition. J. Chem. Phys 2003, 119, 11035–11047.

(23). Grant JA; Pickup BT A Gaussian description of molecular shape. J. Phys. Chem. A 1995, 99, 3503–3510.

(24). Grant JA; Pickup BT; Nicholls A A smooth permittivity function for Poisson–Boltzmann solvation methods. J. Comput. Chem 2001, 22, 608.

(25). Ye X; Wang J; Luo R A revised density function for molecular surface calculation in continuum solvent models. J. Chem. Theory Comput 2010, 6, 1157–1169. [PubMed: 24723844]

(26). Li L; Li C; Zhang Z; Alexov E On the dielectric “constant” of proteins: Smooth dielectric function for macromolecular modeling and its implementation in delphi. J. Chem. Theory Comput 2013, 9, 2126–2136. [PubMed: 23585741]

(27). Chakravorty A; Jia Z; Peng Y; Tajielyato N; Wang L; Alexov E Gaussian-based smooth dielectric function: A surface-free approach for modeling macromolecular binding in solvents. Front. Mol. Biosci 2018, 5, No. 25.

(28). Li C; Jia Z; Chakravorty A; Pahari S; Peng Y; Basu S; Koirala M; Panday SK; Petukh M; Li L; Alexov E Delphi suite: New developments and review of functionalities. J. Comput. Chem 2019, 40, 2502–2508. [PubMed: 31237360]

(29). Hermosilla P; Krone M; Guallar V; Vázquez P-P; Vinacua À; Ropinski T Interactive gpu-based generation of solvent-excluded surfaces. Visual Comput. 2017, 33, 869–881.

(30). Wang J; Luo R Assessment of linear finite-difference Poisson-Boltzmann solvers. J. Comput. Chem 2010, 31, 1689–1698. [PubMed: 20063271]

(31). Cai Q; Hsieh M-J; Wang J; Luo R Performance of nonlinear finite-difference Poisson-Boltzmann solvers. J. Chem. Theory Comput 2010, 6, 203–211. [PubMed: 24723843]

(32). Osher S; Fedkiw R Level Set Methods and Dynamic Implicit Surfaces; Applied Mathematical Sciences Series; Springer: New York, 2003; Vol. 153.

(33). Shrestha A; Mahmood A Review of deep learning algorithms and architectures. IEEE Access 2019, 7, 53040–53065.

(34). Hornik K Approximation capabilities of multilayer feedforward networks. Neural Networks 1991, 4, 251–257.

(35). Gupta A; Wang H; Ganapathiraju M In Learning Structure in Gene Expression Data Using Deep Architectures, with an Application to Gene Clustering, 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM); IEEE, 2015; pp 1328–1335.

(36). Chen L; Cai C; Chen V; Lu X Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model. BMC Bioinf. 2016, 17, No. S9.

(37). Tan J; Hammond JH; Hogan DA; Greene CS Adage-based integration of publicly available pseudomonas aeruginosa gene expression data with denoising autoencoders illuminates microbe-host interactions. mSystems 2016, 1, No. e00025-15.

(38). Tan J; Doing G; Lewis KA; Price CE; Chen KM; Cady KC; Perchuk B; Laub MT; Hogan DA; Greene CS Unsupervised extraction of functional gene expression signatures in the bacterial pathogen Pseudomonas aeruginosa with eadage. bioRxiv 2016, No. 078659.

(39). Chen Y; Li Y; Narayan R; Subramanian A; Xie X Gene expression inference with deep learning. Bioinformatics 2016, 32, 1832–1839. [PubMed: 26873929]

(40). Singh R; Lanchantin J; Robins G; Qi Y Deepchrome: Deep-learning for predicting gene expression from histone modifications. Bioinformatics 2016, 32, i639–i648. [PubMed: 27587684]

(41). Singh R; Lanchantin J; Sekhon A; Qi Y Attend and predict: Understanding gene regulation by selective attention on chromatin. Adv. Neural Inf. Process. Syst 2017, 2017, 6785–6795.

(42). Liang M; Li Z; Chen T; Zeng J Integrative data analysis of multi-platform cancer data with a multimodal deep learning approach. IEEE/ACM Trans. Comput. Biol. Bioinf 2015, 12, 928–937.

(43). Wang S; Sun S; Xu J In Auc-Maximized Deep Convolutional Neural Fields for Protein Sequence Labeling, Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2016; Springer, 2016; pp 1–16.

(44). Jones DT; Singh T; Kosciolek T; Tetchner S Metapsicov: Combining coevolution methods for accurate prediction of contacts and long range hydrogen bonding in proteins. Bioinformatics 2015, 31, 999–1006. [PubMed: 25431331]

(45). Weigt M; White RA; Szurmant H; Hoch JA; Hwa T Identification of direct residue contacts in protein–protein interaction by message passing. Proc. Natl. Acad. Sci. U.S.A 2009, 106, 67–72. [PubMed: 19116270]

(46). Marks DS; Colwell LJ; Sheridan R; Hopf TA; Pagnani A; Zecchina R; Sander C Protein 3d structure computed from evolutionary sequence variation. PLoS One 2011, 6, No. e28766.

(47). Qi Y; Oja M; Weston J; Noble WS A unified multitask architecture for predicting local protein properties. PLoS One 2012, 7, No. e32235.

(48). Heffernan R; Paliwal K; Lyons J; Dehzangi A; Sharma A; Wang J; Sattar A; Yang Y; Zhou Y Improving prediction of secondary structure, local backbone angles and solvent accessible surface area of proteins by iterative deep learning. Sci. Rep 2015, 5, No. 11476.

(49). Jones DT Protein secondary structure prediction based on position-specific scoring matrices. J. Mol. Biol 1999, 292, 195–202. [PubMed: 10493868]

(50). Zhou J; Troyanskaya O In Deep Supervised and Convolutional Generative Stochastic Network for Protein Secondary Structure Prediction, International Conference on Machine Learning, PMLR: 2014, 2014; pp 745–753.

(51). Ma J; Wang S; Wang Z; Xu J Protein contact prediction by integrating joint evolutionary coupling analysis and supervised learning. Bioinformatics 2015, 31, 3506–3513. [PubMed: 26275894]

(52). Di Lena P; Nagata K; Baldi P Deep architectures for protein contact map prediction. Bioinformatics 2012, 28, 2449–2457. [PubMed: 22847931]

(53). Eickholt J; Cheng J Predicting protein residue–residue contacts using deep networks and boosting. Bioinformatics 2012, 28, 3066–3072. [PubMed: 23047561]

(54). Skwark MJ; Raimondi D; Michel M; Elofsson A Improved contact predictions using the recognition of protein like contact patterns. PLoS Comput. Biol 2014, 10, No. e1003889.

(55). AlQuraishi M End-to-end differentiable learning of protein structure. Cell Syst. 2019, 8, 292.e3–301.e3. [PubMed: 31005579]

(56). De Las Rivas J; Fontanillo C Protein–protein interactions essentials: Key concepts to building and analyzing interactome networks. PLoS Comput. Biol 2010, 6, No. e1000807.

(57). Zhou D; He Y Extracting interactions between proteins from the literature. J. Biomed. Inf 2008, 41, 393–407.

(58). Peng Y; Lu Z Deep Learning for Extracting Protein–Protein Interactions from Biomedical Literature. 2017, arXiv:1706.01556. arXiv.org e-Print archive. https://arxiv.org/abs/1706.01556.

(59). Du X; Sun S; Hu C; Yao Y; Yan Y; Zhang Y Deepppi: Boosting prediction of protein–protein interactions with deep neural networks. J. Chem. Inf. Model 2017, 57, 1499–1510. [PubMed: 28514151]

(60). Sun T; Zhou B; Lai L; Pei J Sequence-based prediction of protein protein interaction using a deep-learning algorithm. BMC Bioinf. 2017, 18, No. 277.

(61). Wang Y-B; You Z-H; Li X; Jiang T-H; Chen X; Zhou X; Wang L Predicting protein–protein interactions from protein sequences by a stacked sparse autoencoder deep neural network. Mol. BioSyst 2017, 13, 1336–1344. [PubMed: 28604872]

(62). Du T; Liao L; Wu CH; Sun B Prediction of residue-residue contact matrix for protein-protein interaction with fisher score features and deep learning. Methods 2016, 110, 97–105. [PubMed: 27282356]

(63). Ching T; Himmelstein DS; Beaulieu-Jones BK; Kalinin AA; Do BT; Way GP; Ferrero E; Agapow P-M; Zietz M; Hoffman MM; et al. Opportunities and obstacles for deep learning in biology and medicine. J. R. Soc., Interface 2018, 15, No. 20170387.

(64). Davis ME; Mccammon JA Dielectric boundary smoothing in finite-difference solutions of the poisson equation - an approach to improve accuracy and convergence. J. Comput. Chem 1991, 12, 909–912.

(65). Wang J; Cai Q; Xiang Y; Luo R Reducing grid dependence in finite-difference Poisson-Boltzmann calculations. J. Chem. Theory Comput 2012, 8, 2741–2751. [PubMed: 23185142]

(66). Wei H; Luo A; Qiu T; Luo R; Qi R Improved Poisson–Boltzmann methods for high-performance computing. J. Chem. Theory Comput 2019, 15, 6190–6202. [PubMed: 31525962]

(67). Li Z; Ito K The Immersed Interface Method: Numerical Solutions of PDEs Involving Interfaces and Irregular Domains; SIAM, 2006; Vol. 33.

(68). Wang J; Cai Q; Li Z-L; Zhao H-K; Luo R Achieving energy conservation in Poisson–Boltzmann molecular dynamics: Accuracy and precision with finite-difference algorithms. Chem. Phys. Lett 2009, 468, 112–118. [PubMed: 20098487]

(69). Wang C; Wang J; Cai Q; Li Z; Zhao H-K; Luo R Exploring accurate Poisson–Boltzmann methods for biomolecular simulations. Comput. Theor. Chem 2013, 1024, 34–44. [PubMed: 24443709]

(70). Wei H; Luo R; Qi R An efficient second-order Poisson–Boltzmann method. J. Comput. Chem 2019, 1257.

(71). Emmert-Streib F; Yang Z; Feng H; Tripathi S; Dehmer M An introductory review of deep learning for prediction models with big data. Front. Artif. Intell 2020, 3, No. 4.

(72). Case DA; Belfon K; Ben-Shalom IY; Brozell SR; Cerutti DS; Cheatham TE; Cruzeiro VWD; Darden TA; Duke RE; Giambasu G; Gilson MK; Gohlke H; Goetz AW; Harris R; Izadi S; Izmailov SA; Kasavajhala K; Kovalenko A; Krasny R; Kurtzman T; Lee TS; LeGrand S; Li P; Lin C; Liu J; Luchko T; Luo R; Man V; Merz KM; Miao Y; Mikhailovskii O; Monard G; Nguyen H; Onufriev A; Pan F; Pantano S; Qi R; Roe DR; Roitberg A; Sagui C; Schott-Verdugo S; Shen J; Simmerling CL; Skrynnikov NR; Smith J; Swails J; Walker RC; Wang J; Wilson L; Wolf RM; Wu X; Xiong Y; Xue Y; York DM; Kollman PA Amber 2020; University of California: San Francisco, 2020.

(73). Chollet F Keras; Github, 2015.

(74). Cai Q; Ye X; Wang J; Luo R On-the-fly numerical surface integration for finite-difference Poisson-Boltzmann methods. J. Chem. Theory Comput 2011, 7, 3608–3619. [PubMed: 24772042]

(75). Qi R; Botello-Smith WM; Luo R Acceleration of linear finite-difference Poisson–Boltzmann methods on graphics processing units. J. Chem. Theory Comput 2017, 13, 3378–3387. [PubMed: 28553983]

(76). Qi R; Luo R Robustness and efficiency of Poisson–Boltzmann modeling on graphics processing units. J. Chem. Inf. Model 2019, 59, 409–420. [PubMed: 30550277]

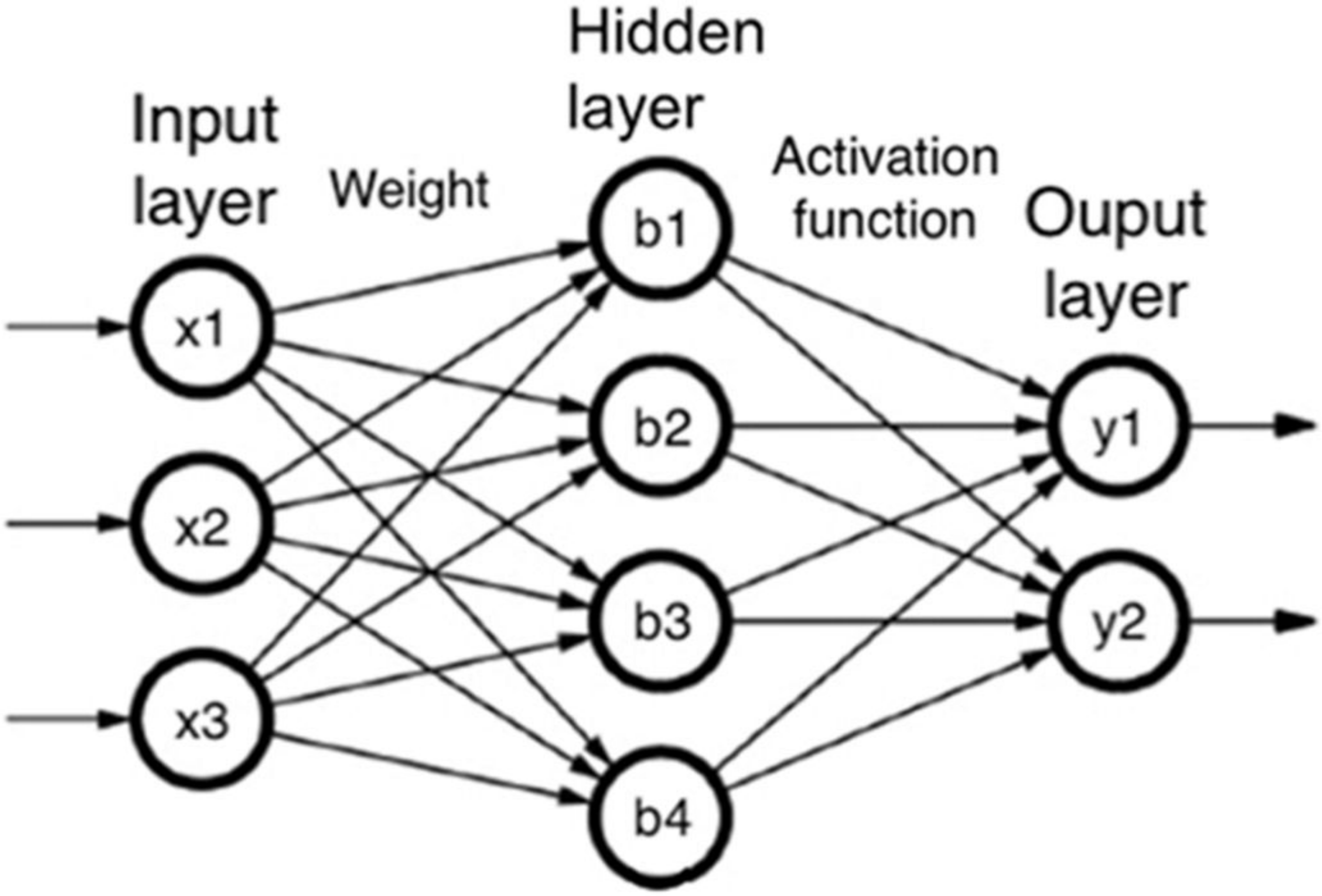


Figure 1.
Structure of an artificial neural network with one hidden layer. Here, each circular node represents an artificial neuron, and an arrow represents a connection from the output of one artificial neuron to the input of another.

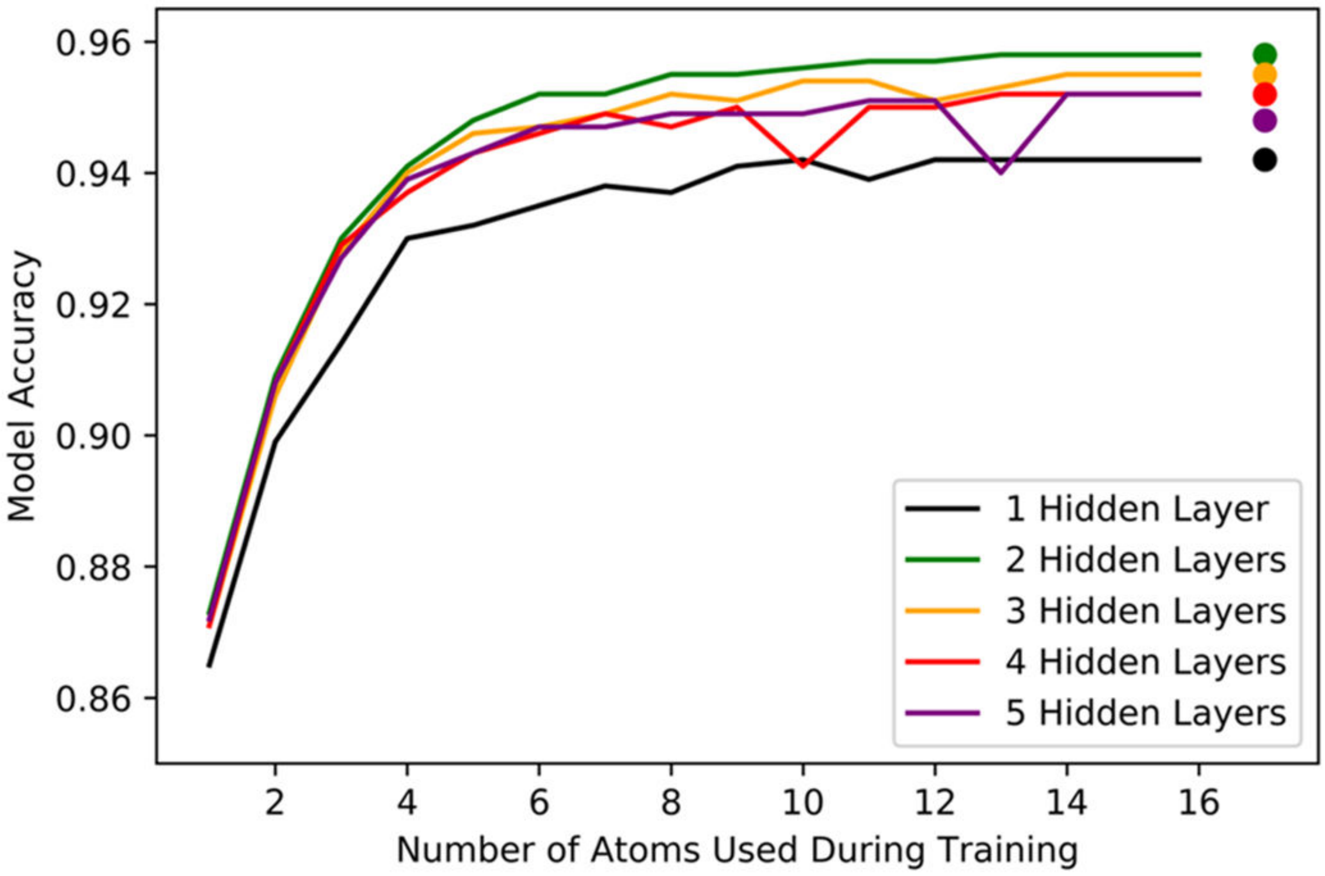


Figure 2.

Incremental training of the model. The ANN model was set up with different numbers of hidden layers; each hidden layer is of 200 neurons. The model was trained with the first training data set. The analysis was also repeated with two additional training sets, and the figures are shown in Figure S2, Supporting Information.

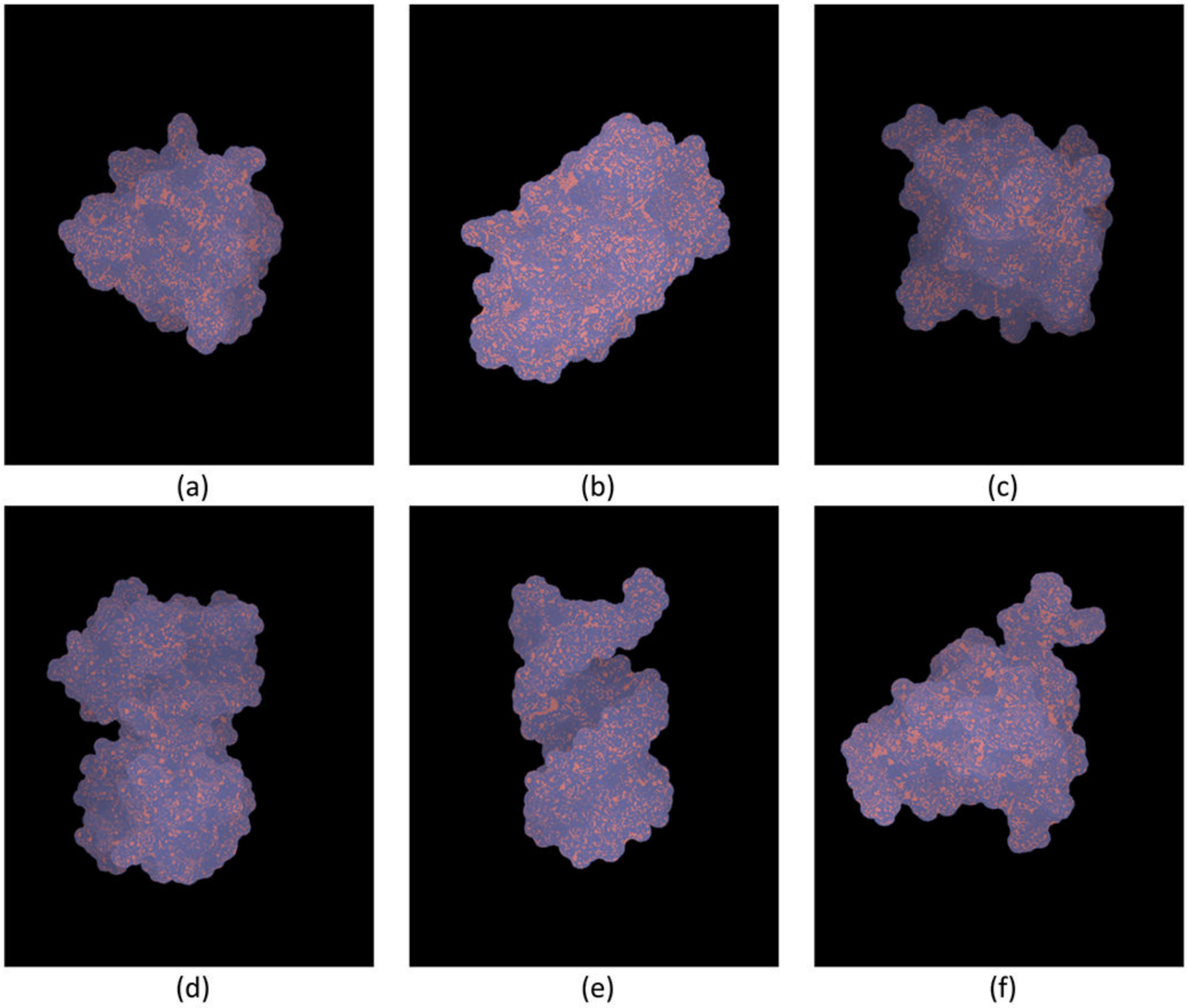


Figure 3.

Superimposed rendering of the machine-learned SES (blue) and the classical SES (red) of representative molecules. (a) PDB ID: 1enh, all-α protein; (b) PDB ID: 1pgb, all-β protein; (c) PDB ID: 1shg, α/β protein; (d) PDB ID: 1w0u, protein/protein complex; (e) PDB ID: 3czw, RNA duplex; and (f) PDB ID: 3fdt, protein/DNA complex.

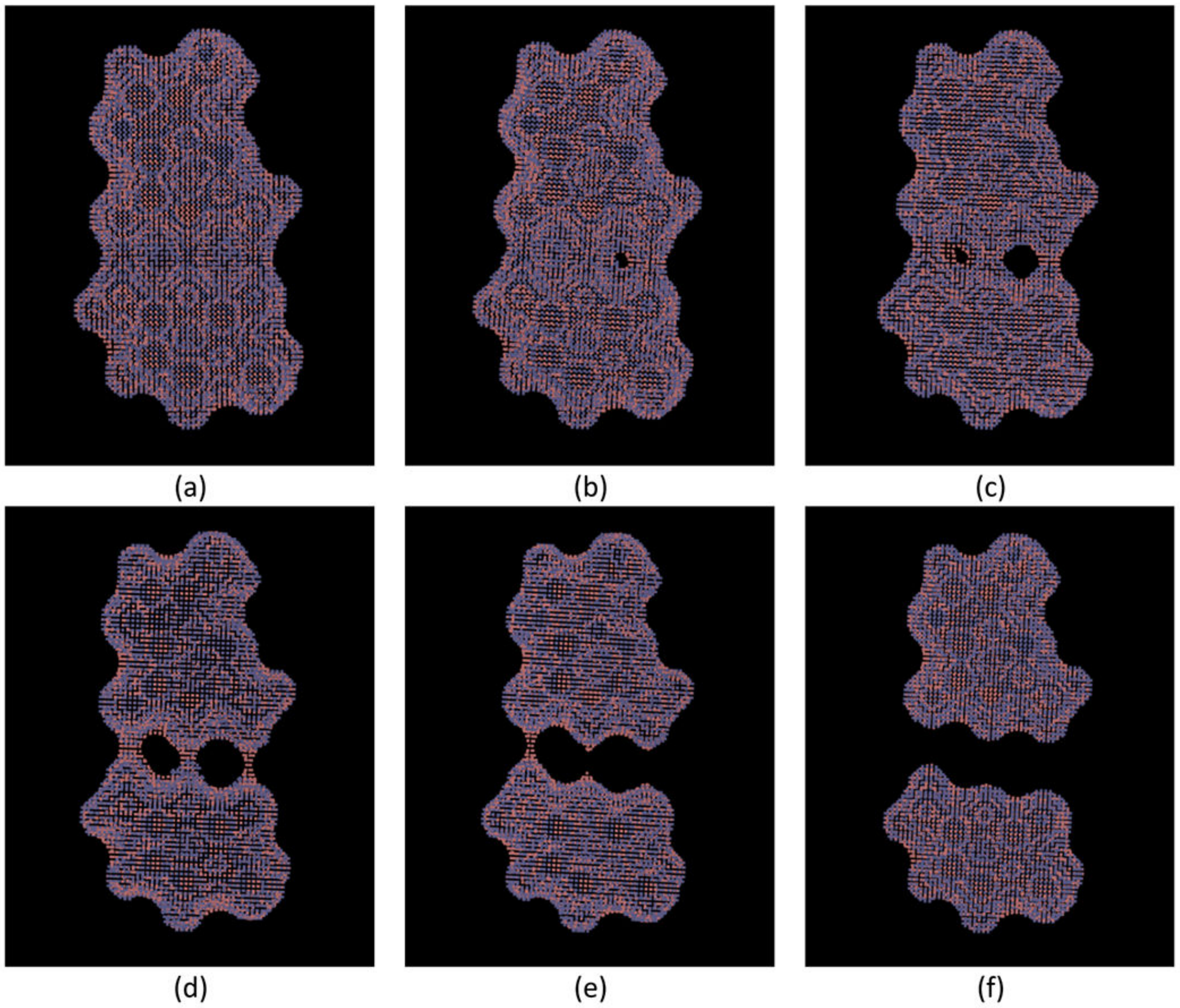


Figure 4.

Superimposed rendering of machine-learned SES (blue) and the classical SES (red) of the GC complex. Starting from the hydrogen-bonded structure, the two molecules are manually pulled apart for 0.5 Å each step. The standalone figures of the two surfaces are included as Figures S8 and S9 in the Supporting Information.

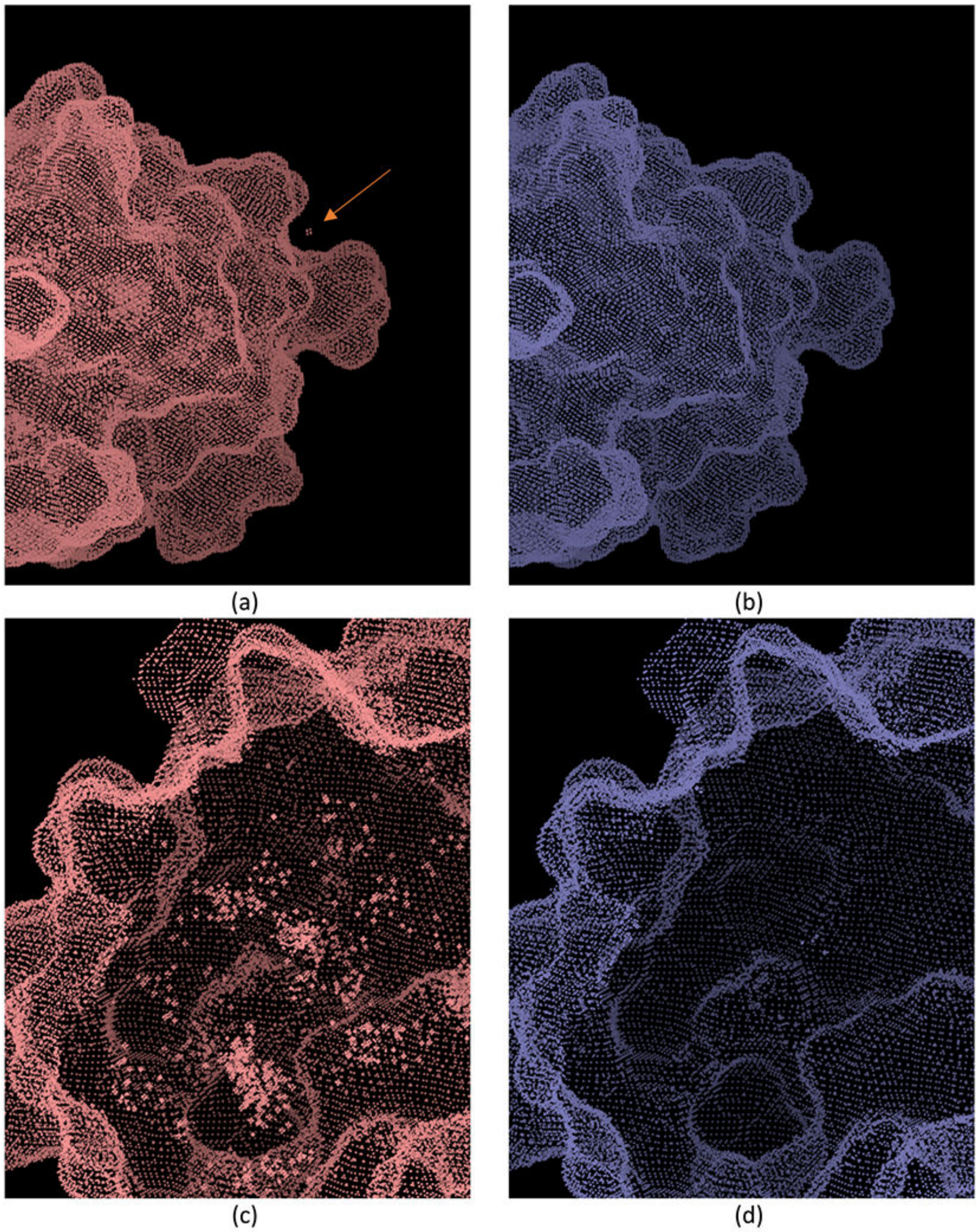


Figure 5.

Detailed views of SES as predicted by the first and second versions of the machine-learned SES model. The PDB id of the tested molecule is 1shg. Panels (a) and (c) are from the first version of the model, outside vision and the inside vision, respectively. Panels (b) and (d) are from a second version of the model, outside vision and the inside vision, respectively.

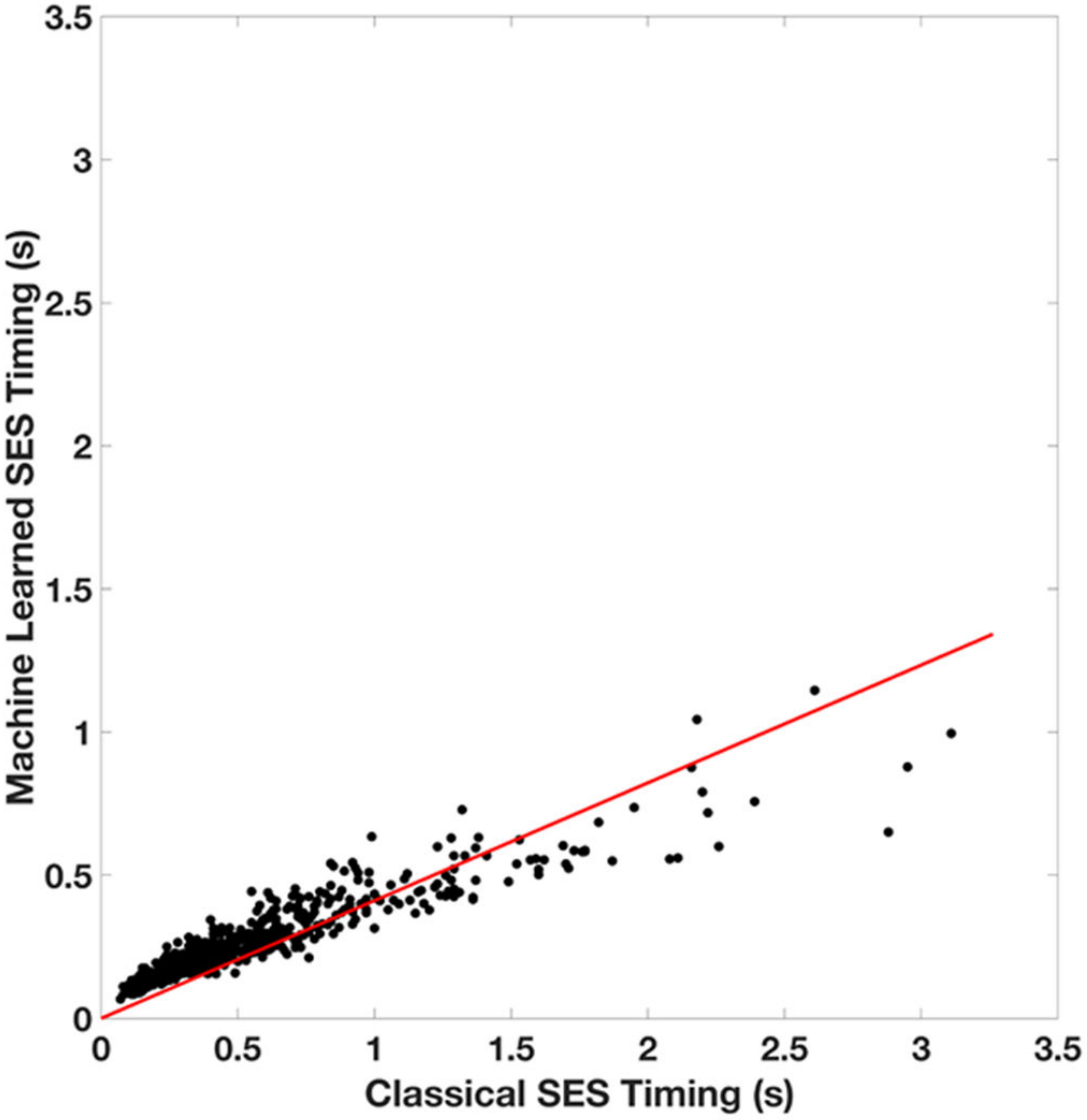


Figure 6.
Timing comparison of the machine-learned SES and the classical SES. Both sets of data were collected on a single core of an INTEL Xeon E5-4620 CPU. The grid spacing was set to 0.95 Å as used in the collection of training data. The regression line between the two sets of timing data is $y = 0.4114x$.

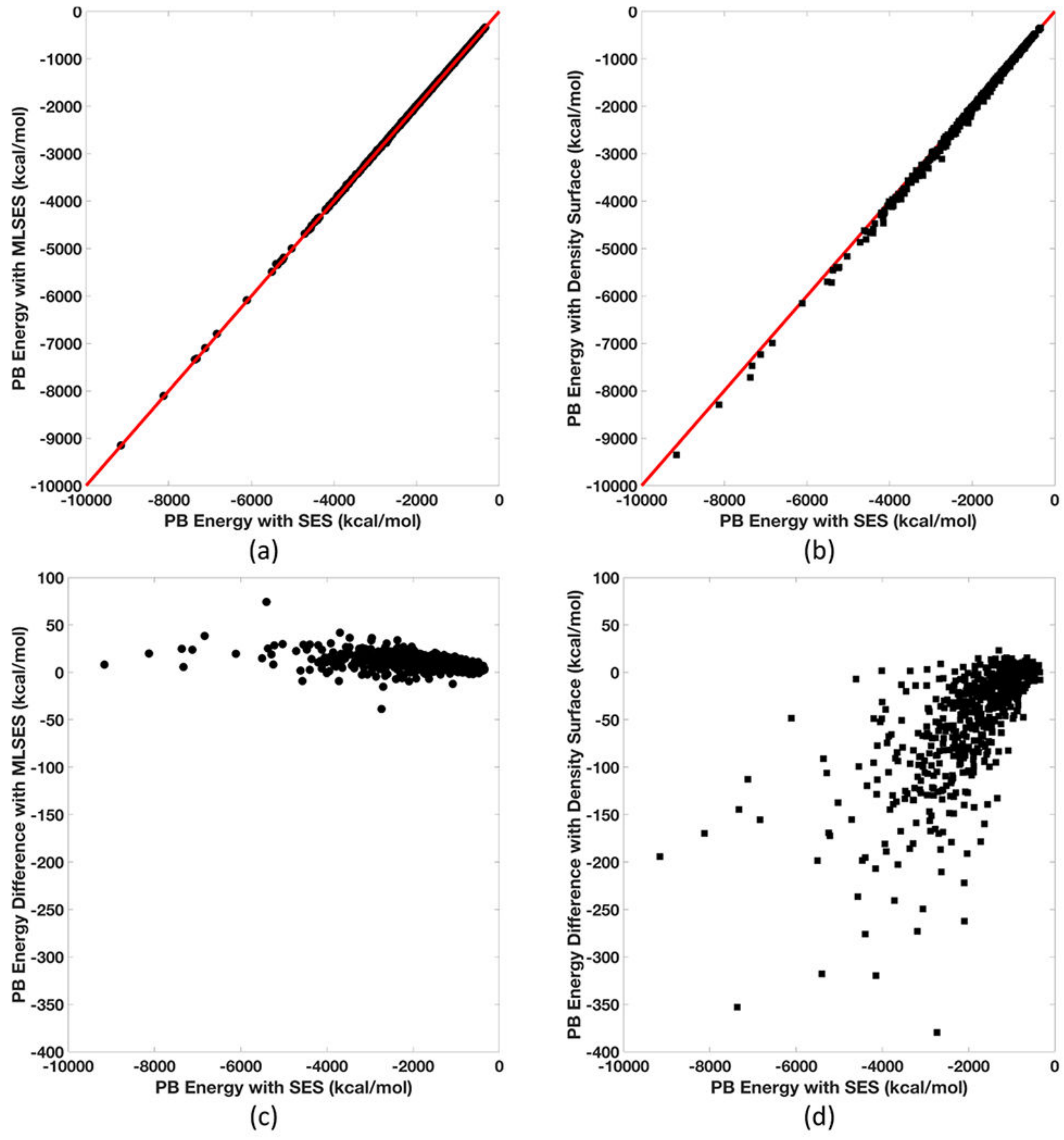


Figure 7.

PB reaction field energies with different molecular surfaces. (a) Correlation between energies from machine-learned SES and classical SES. (b) Correlation between energies from density function and classical SES. (c) Energy differences in panel (a). (d) Energy differences in panel (b). The lines in panels (a) and (b) are the diagonal line "$y = x$" as reference.

Table 1.

Model Accuracy with Different Structural Data Sets[a]

data sets	training data 1	training data 2	training data 3	training data 4	training data 5	training data 6	nucleic acid	protein complex
accuracy (%)	95.96	96.01	96.02	96.03	96.26	97.78	97.07	95.16

[a]The first six data sets are those from the training phase, and the last two sets are not included in the training phase.

Table 2.

Rotational Symmetry Test[a]

angle	**30**	**60**	**90**	**120**	**150**	**180**
rate (%)	98.6	98.5	98.1	98.3	98.8	98.4
angle	**210**	**240**		**270**	**300**	**330**
rate (%)	98.2	98.8		98.7	98.6	99.1

[a]The agreement rate is the percentage of transformed inputs that give the same prediction (+1 or −1) as the original inputs.