Automation of (Macro)molecular Properties Using a Bootstrapping Swarm Artificial Neural Network Method: Databases for Machine Learning

Blerta Rahmani[†] and Hiqmet Kamberaj[*,‡,†,¶]

[†]*Department of Physics, Faculty of Mathematics and Natural Sciences, State University of Tetova, Tetova, North Republic of Macedonia*

[‡]*Advanced Computing Research Center, University of New York Tirana, Tirana, Albania*

[¶]*Department of Computer Engineering, Faculty of Engineering, International Balkan University, Skopje, North Republic of Macedonia*

E-mail: h.kamberaj@gmail.com

Abstract

In this study, we employed a novel method for prediction of (macro)molecular properties using a swarm artificial neural network method as a machine learning approach. In this method, a (macro)molecular structure is represented by a so-called *description vector*, which then is the input in a so-called *bootstrapping swarm artificial neural network* (BSANN) for training the neural network. In this study, we aim to develop an efficient approach for performing the training of an artificial neural network using either experimental or quantum mechanics data. In particular, we aim to create different user-friendly online accessible databases of well-selected experimental (or quantum mechanics) results that can be used as proof of the concepts. Furthermore, with the optimized artificial neural network using the training data served as input for BSANN, we can predict properties and their statistical errors of new molecules using the plugins provided from that web-service. There are four databases accessible using the web-based service. That includes a database of 642 small organic molecules with known experimental hydration free energies, the database of 1475 experimental pKa values of ionizable groups in 192 proteins, the database of 2693 mutants in 14 proteins with given values of experimental values of changes in the Gibbs free energy, and a database of 7101 quantum mechanics heat of formation calculations.

All the data are prepared and optimized in advance using the AMBER force field in CHARMM macromolecular computer simulation program. The BSANN is code for performing the optimization and prediction written in Python computer programming language. The descriptor vectors of the small molecules are based on the Coulomb matrix and sum over bonds properties, and for the macromolecular systems, they take into account the chemical-physical fingerprints of the region in the vicinity of each amino acid.

1 Introduction

Recently, neural network method has seen a broad range of applications in molecular modeling.[1] In Ref.,[2] a hierarchical interacting particle neural network approach is introduced using quantum models to predict molecular properties. In this approach, different hierarchical regularization terms have been introduced to improve the convergence of the optimized parameters. While in Ref.,[3] the machine learning like-potentials are used to predict molecular properties, such as enthalpies or potential energies. The degree to which the general features included in characterizing the chemical space of molecules to improve the predictions of these models is also discussed in Refs.[4,5] Tuckerman and co-workers[6] used a stochastic neural network technique to fit high-dimensional free energy surfaces characterized by reduced subspace of collective coordinates. While very recently[7] a comparison study has been performed between neural network approach and Gaussian process regression to fit the potential energy surfaces. One of the recognized problems in using machine learning approaches in prediction free energy surfaces is the inaccurate representation of general features of the surface topology by the training data. To improve on this, a combination of metadynamics molecular dynamics with neural network chemical models have also proposed.[8] It is worth noting that in the prediction of free energy surfaces, an accurate representation of the reduced subspace can be important. For that, Wehmeyer & Noé[9] have used the time-lagged auto-encoder to determine essential degrees of freedom of dynamical data.

Machine learning approaches have also been in the field of drug-design, for instance, in predicting drug-target interactions,[10] and it is a promising approach. In particular, the method is used in combination with molecular dynamics to predict the ligand-binding mechanism to purine nucleoside phosphorylase,[11] and it accurately identifies the mechanism of drug-target binding modes.

In this study, we employed a novel method for prediction of (macro)molecular properties using a swarm artificial neural network method as a machine learning approach. In this method, a (macro)molecular structure is represented by a so-called *description vector*, which then is used as input in a so-called *bootstrapping swarm artificial neural network* (BSANN) for training the neural network. We aim to develop an efficient approach for performing the training of an artificial neural network using either experimental or quantum mechanics data. In particular, we created different user-friendly online accessible databases of well-selected experimental (or quantum mechanics) results that can be used as proof of the concepts. Furthermore, with the optimized artificial neural network using the training data served as input for BSANN, we can predict properties and their statistical errors of new molecules using the plugins provided from that web-service.

There are four databases accessible using the web-based service. The database of 642 small organic molecules with known experimental hydration free energies,[12] which well-studied in Ref.;[13] the database of 1475 experimental pKa values of ionizable groups in 192 proteins (including 153 wild-type proteins and 39 mutant proteins);[14–17] the database of 2693 mutants in 14 proteins with given values of experimental values of changes in the Gibbs free energy;[18,19] and a database of 7101 quantum mechanics heat of formation calculations with the Perdew-Burke-Ernzerhof hybrid functional (PBE0).[5,20]

All the data are prepared and optimized in advance using the AMBER force field[21] in CHARMM macromolecular computer simulation program.[22] The BSANN is the code for performing the optimization and prediction written in Python computer programming language. The descriptor vectors of the small molecules are based on the Coulomb matrix and the sum over bonds properties, and for the macromolecular systems they take into account the chemical-physical fingerprints of the region in the vicinity of each amino acid.

2 Materials and Methods

2.1 Artificial Neural Network

Machine Learning (ML) approach provides a potential method to predict the properties of a system using decision-making algorithms, based on some predefined features characterizing these properties of the system. There exist different ML methods used to predict missing data or discover new patterns during the data mining process.[23] Neural networks method considers a large training dataset, and then it tries to construct a system, which is made up of rules for recognizing the patterns within the training data set by a learning process.

In general, for an ANN with K hidden layers (see also Figure 1), the output Y_i is defined as

$$Y_i = f\left(\underbrace{\sum_{l_K=1}^{L_K} f\left(\underbrace{\sum_{l_{K-1}=1}^{L_{K-1}} f\left(\underbrace{\cdots f\left(\underbrace{\sum_{l_2=1}^{L_2} f\left(\underbrace{\sum_{l_1=1}^{L_1} \underbrace{f\left(\sum_{j=1}^{n} X_j W_{jl_1} + b_{l_1}\right)}_{\text{input layer}} W_{l_1 l_2} + b_{l_2}\right)}_{\text{1st hidden layer}} \cdots \right)}_{\text{2nd hidden layer}} \cdots \right)}_{\vdots} W_{l_{K-1}l_K} + b_{l_K} \right)}_{(K-1)\text{th hidden layer}} W_{l_K i} + b_i \right)}_{K\text{th hidden layer}} \quad (1)$$

Here, $\mathbf{W}$ and $\mathbf{b}$ are considered as free parameters, which need to be optimized for a given training data used as inputs and given outputs, which are known. To optimize these parameters the so-called loss function is minimized using Gradient Descent method:[25]

$$S(\mathbf{W}, \mathbf{b}) = \sum_{i=1}^{m} \left(Y_i^0 - Y_i\right)^2 \quad (2)$$

where $\mathbf{Y}^0$ represent the true output vector. For that, the gradients of $S(\mathbf{W}, \mathbf{b})$ with respect

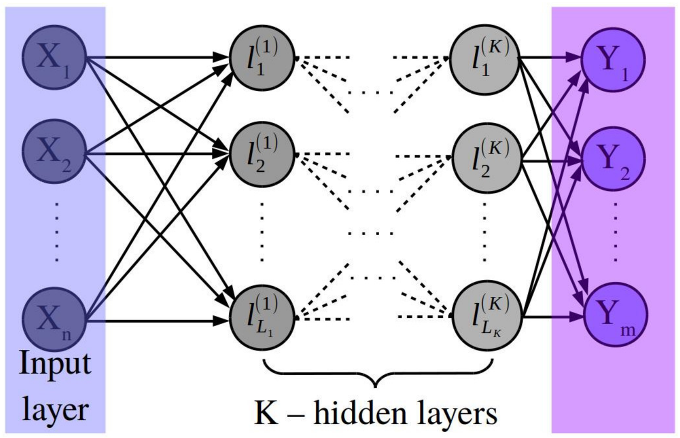


Figure 1: Illustration diagram of an artificial neural network (ANN). It is characterized by an input vector of dimension n, K hidden layers of $l_{L_1}^{(1)}, l_{L_2}^{(2)}, \cdots, l_{L_K}^{(K)}$ neurons each, and an output vector of dimension m.[24]

to $\mathbf{W}$ and $\mathbf{b}$ are calculated:[25]

$$\Delta\mathbf{W} = -\left(\frac{\partial S(\mathbf{W},\mathbf{b})}{\partial \mathbf{W}}\right)_{\mathbf{b}} \qquad (3)$$
$$\Delta\mathbf{b} = -\left(\frac{\partial S(\mathbf{W},\mathbf{b})}{\partial \mathbf{b}}\right)_{\mathbf{W}}$$

To avoid over-fitting, which is one of pitfalls of the machine learning approaches,[26] the following regularization terms have been introduced in literature:

$$\Delta'\mathbf{W} = \gamma_w\left(\Delta\mathbf{W} + \gamma_1\mathbf{W}\right) \qquad (4)$$
$$\Delta'\mathbf{b} = \gamma_w\left(\Delta\mathbf{b} + \gamma_1\mathbf{b}\right)$$

where γ_w is called learning rate for the gradient and γ_1 is called the regulation strength.

Usually, the Gradient Descent method often converges to a local minimum, and hence it provides a local optimization to the problem. To avoid that a new Bootstrapping Swarm Artificial Neural Network method is proposed in the literature,[24] which is introduced in the following.

2.2 Bootstrapping Swarm Artificial Neural Network

The standard ANN method deals with random numbers, which are used to initialize the parameters $\mathbf{W}$ and $\mathbf{b}$; therefore, the optimal solution of the problem will be different for different runs. In particular, we can say that there exists an uncertainty in the calculation of the optimal solution (i.e., in determining $\mathbf{W}$ and $\mathbf{b}$.) To calculate these uncertainties in the calculation of the optimal parameters, $\mathbf{W}$ and $\mathbf{b}$, we introduce a new approach, namely bootstrapping artificial neural network based on the method proposed by Gerhard Paass,[27] or similar methods.[28] In this approach, M copies of the same neural network are run independently using different input vectors. Here, we implement that at regular intervals, to swap optimal parameters (i.e., $\mathbf{W}$ and $\mathbf{b}$) between the two neighboring neural networks, which is equivalent to increasing the dimensionality of the problem by one; that is, if the dimensionality in each of the replicas is $d = K \times L$, then the dimensionality of the bootstrapping artificial neural network based on the method is $d + 1$. Figure 2 shows the layout of this configuration.

Furthermore, to achieve a good sampling of the phase space extended by the vectors $\mathbf{W}$ and $\mathbf{b}$, we introduce two other regularization terms similar to the swarm-particle sampling approach. First, we define two vectors for each neural network, namely $\mathbf{W}_n^{\text{Lbest}}$ and $\mathbf{b}_n^{\text{Lbest}}$, which represent the best local optimal parameters for each neural network n. In addition, we also define $\mathbf{W}^{\text{Gbest}}$ and $\mathbf{b}^{\text{Gbest}}$, which represent the global best optimal parameters among all neural networks.

Then, the expressions in Eq. 4 are modified by introducing these two regularization terms as the following:

$$\begin{aligned}\Delta''\mathbf{W}_n &= \gamma_w \left(\Delta\mathbf{W}_n + \gamma_1\mathbf{W}_n \right.\\ &\left. - \gamma_2 U(0,1)\left(\mathbf{W}_n - \mathbf{W}_n^{\text{Lbest}}\right) - \gamma_3 U(0,1)\left(\mathbf{W}_n - \mathbf{W}^{\text{Gbest}}\right)\right)\\ \Delta''\mathbf{b}_n &= \gamma_w \left(\Delta\mathbf{b}_n + \gamma_1\mathbf{b}_n \right.\\ &\left. - \gamma_2 U(0,1)\left(\mathbf{b}_n - \mathbf{b}_n^{\text{Lbest}}\right) - \gamma_3 U(0,1)\left(\mathbf{b}_n - \mathbf{b}^{\text{Gbest}}\right)\right)\end{aligned} \tag{5}$$

for each neural network configuration n, $n = 1, 2, \cdots, M$. Here, $U(0,1)$ is a random number between zero and one, and γ_2 and γ_3 represent the strength of biases toward the local best optimal parameters and global best optimal parameters, respectively. The first term indicates the individual knowledge of each neural network and the second bias term the social knowledge among the neural networks. This method is called here, Bootstrapping Swarm Artificial Neural Network (BSANN). Then, the weights, $\mathbf{W}_n$, and biases, $\mathbf{b}_n$, for each neural network n are updated at each iteration step according to:

$$\begin{aligned}\mathbf{W}_n^{\text{new}} &= \mathbf{W}_n^{\text{old}} + \Delta''\mathbf{W}_n\\ \mathbf{b}_n^{\text{new}} &= \mathbf{b}_n^{\text{old}} + \Delta''\mathbf{b}_n\end{aligned} \tag{6}$$

2.3 (Macro)molecular Feature Description

To construct data-driven models, such as in the ML approach, we will need to specify a list of physical and chemical properties of the input (macro)molecule that contain necessary information about the system. Here, the input data will be presented by a vector of length N, called $\mathbf{X}$. That process is called *feature description*, and the input data are called *feature descriptors*.

Often, a so-called *Simplified Molecular Input Line Entry System* (SMILES) is used to represent a small molecule as a string of letters.[29] In such case, the atoms could be encoded

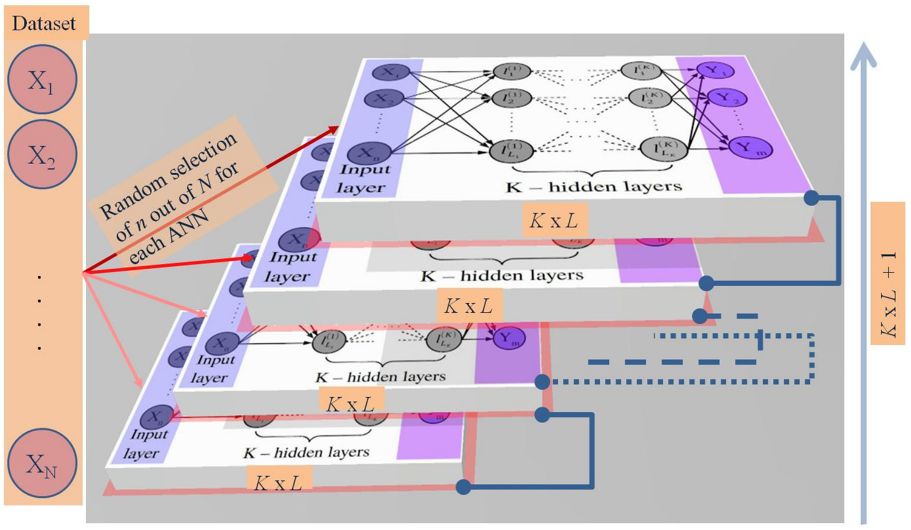


Figure 2: Layout of the Bootstrapping Swarm Artificial Neural Network (BSANN) as adopted by.[24] It is characterized by M different input vectors each of dimension n, K hidden layers of $l_{L_1}^{(1)}$, $l_{L_2}^{(2)}$, $\cdots$, $l_{L_K}^{(K)}$ neurons each, and M different output vectors each of dimension m. Every two neighboring neural networks communicate regularly with each other by swapping the optimized parameters.

by a single integer number, such as H= 1, C= 2, N= 3, and so on, or by the nuclear charge Z, such as H= 1, C= 6, N= 7, and so on.[30] It can be seen that it creates an (unnecessary) relationship between the input data, namely H<C<N, which could influence on the network performance. Other encoding models are also suggested, for instance, representing each atom of the input molecule by the following fingerprint: H= $[1\ 0\ 0\ \cdots]$, C= $[0\ 1\ 0\ \cdots]$, N= $[0\ 0\ 1\ \cdots]$, and so on.[30] However, these fingerprints do also have drawbacks because the dimensions of the encoding vector depend on the number of atoms in the structure and may vary from molecule to molecule; also, based on this model, the atoms belonging to the same group in periodic table of elements do not behave the same.

In this study, we used the so-called *Coulomb matrix*, $\mathbf{C}$, to encode the molecular features, which contains both the geometrical information of the three-dimensional structure and the atom type in the molecule.[20] For any two atoms i and j in a given input molecule, the matrix element C_{ij} is defined as:

$$C_{ij} = \begin{cases} \dfrac{Z_i^{2.4}}{2}, & i = j \\ \dfrac{Z_i Z_j}{r_{ij}}, & i \neq j \end{cases} \tag{7}$$

where Z_i is the atomic number of the ith atom and r_{ij} is the distance between the atoms i and j. The fingerprint represented by the Coulomb matrix, $\mathbf{C}$, has some advantages, such as it takes into account the three-dimensional molecular structure, and it is invariant under rotational and translation of the structure. To calculate $\mathbf{C}$ for a given molecular structure, we need the nuclear charges for each atom and the Cartesian coordinates of the atomic positions taken from the equilibrium geometry. However, note that $\mathbf{C}$ is not invariant under the permutations of the atom order in molecule. Therefore, the spectrum of eigenvalues of matrix $\mathbf{C}$ can be used as a fingerprint of the molecule, since they are invariant under both rotation/translation and permutations of the rows and columns. A second feature descriptor that we used in this study is the so-called *sum over bonds*, which is a numerical descriptor representing the vector of bond types present in a molecule, similar to.[31] If N_b is the number of unique bonds in the dataset of the compounds studied, then a vector with dimensions N_b is constructed for each molecule with entry either zeros or the integers giving the frequency of appearance for each bond type in molecular structure. This fingerprint descriptor vector has a unique length within the dataset. Then, the vector descriptor of the sum over bonds is concatenated at the end of the Coulomb matrix descriptor. To construct the input descriptor vector for a macromolecule, we introduced the following model. For example, suppose we would like to calculate the change on the Gibbs free energy upon the mutations (either single or multiple mutations) or perform pKa calculations for a selected residue in a protein. We label each residue or nucleotide of the input sequence with an ID from 1 to 24. That is, we form a descriptor vector with length $N_1 = 24$, $\mathbf{X}_1$, which is a vector of zeros and ones defined as the following:

$$\begin{array}{cccccccc}
 & & \text{VAL} & & \cdots & \text{THR} & \cdots & \leftarrow \text{mutation point}\\
 & & \downarrow & & \cdots & \downarrow & \cdots & \\
\mathbf{X}_1 = & 0 & 1 & 0 & \cdots & 1 & \cdots & \leftarrow \text{descriptor vector}\\
 & \uparrow & \uparrow & \uparrow & \cdots & \uparrow & \cdots & \\
 & \text{ALA} & \text{VAL} & \text{LEU} & \cdots & \text{THR} & \cdots & \leftarrow \text{A. A. dictionary}
\end{array}$$

where *A. A. dictionary* represents the dictionary of the all amino acids. In addition, to characterize the environment around any mutation point, we determine another descriptor vector, namely $\mathbf{X}_2$ with length $N_2 = 24$, which is defined as the following. For each mutation point amino acid i, we determine the nearest neighbor amino acids $k = i_1, i_2, \cdots, i_{n.n.}$, based, for example, on the center of mass distance. Then, the jth element $X_j^{(2)}$ of the vector $\mathbf{X}_2$ is defined as a modified 'Coulombic matrix':

$$X_j^{(2)} = \sum_i \sum_{k=i_1}^{i_{n.n.}} \begin{cases} \dfrac{1}{r_{ik}}, & k = j \\ 0, & k \neq j \end{cases} \tag{8}$$

where the first sum runs over all point mutation amino acids, and the second sum runs over all nearest neighbors of amino acids i. In Eq. 11, r_{ik} denotes the center-to-center distance between the two amino acids. To take into account the polarity of the amino acids, we introduce a binary vector of dimension $N_p = 3$, such that

$$\mathbf{X}_p^{(1)} = [1\ 0\ 0]\,, \quad \mathbf{X}_p^{(2)} = [0\ 1\ 0]\,, \quad \mathbf{X}_p^{(3)} = [0\ 0\ 1] \tag{9}$$

where $\mathbf{X}^{(1)}$ represents a non-polar amino acid, $\mathbf{X}^{(2)}$ represents an uncharged polar amino acid, and $\mathbf{X}^{(3)}$ represents a charged polar amino acid. In addition, we also added another component to the net vector, which is a real value representing the percentage of the buried part of the amino acids ($\%\text{SASA}_{\text{buried}}$), which is defined as the ratio of the buried surface with the solvent accessible surface area of the amino acid in the protein structure, and it is represented by the vector $\mathbf{X}_4$. Note that vector $\mathbf{X}_4$ can also include other properties, such as the temperature, concentration of the salt and pH value of the experiment; therefore, we can write:

$$\mathbf{X}_4 = [\%\mathrm{SASA}_{\mathrm{buried}}\ T\ c\ \mathrm{pH}\ \cdots] \tag{10}$$

where T, c, and pH are the temperature (in kelvin), concentration (in molar) and pH, respectively.

To determine the descriptor vector of the macromolecule, such as protein, we concatenate the vectors $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_p^{(i)}$ and $\mathbf{X}_4$ into a net descriptor vector $\mathbf{X}$ with length $N = 55$. Note that in the expression given by Eq. 11, other properties can be encoded. For example, we can encode the dielectric properties in the vicinity of each amino acid in the structure by modifying Eq. 11 as follows:

$$X_j^{(2)} = \sum_i \sum_{k=i_1}^{i_{n.n.}} \begin{cases} \dfrac{1}{\varepsilon_i r_{ik}}, & k = j \\ 0, & k \neq j \end{cases} \tag{11}$$

where ε_{ik} is the dielectric constant of the environment in the vicinity of the mutated amino acid i, which can be taken a simple distant dependent dielectric constant between the amino acid i and its nearest neighbor k: $\varepsilon_{ik} = Dr_{ik}$, where D is a constant, or even other complicated distance dependence functions.[32,33] However, in this work, other more complicated distance dependent dielectric constant is considered, such as the sigmoidal function:[32,33]

$$\varepsilon(r) = \frac{\varepsilon_w + D_0}{1 + k\exp\left(-\kappa\left(\varepsilon_w + D_0\right) r\right)} - D_0 \tag{12}$$

where r is the distance between two amino acids, $\varepsilon_w = 80$ is the dielectric constant of water, $D_0 = 8$, $\kappa = 0.5/(\varepsilon_w + D_0)$, $k = \left(\varepsilon_w - \varepsilon_p\right)/\left(D_0 + \varepsilon_p\right)$ with $\varepsilon_p = 2$ being the dielectric constant of protein. A plot of the $\varepsilon(r)$ versus the distance r is presented in Figure 3 for both simple function of the distance of dielectric constant and sigmoidal distance dependence function of the dielectric constant. Here, sigmoidal function gives a smooth variation of the dielectric constant screening the electrostatic interactions from 2 (which is the dielectric constant of the internal protein) to 80 (which is the dielectric constant of bulk water, as shown in Figure 3.

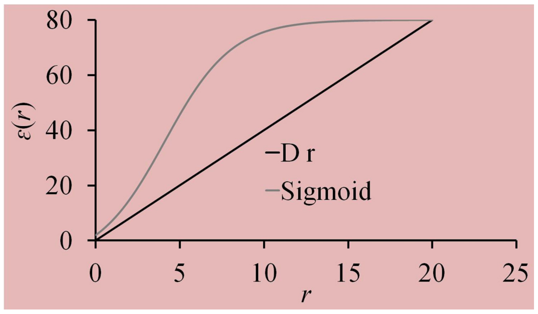


Figure 3: Dielectric constants as a function of the distance r between the amino acids for the two cases: $\varepsilon = Dr$ with $D = 8$, and dielectric constant given by Eq. 12 for $\varepsilon_w = 80$ is the dielectric constant of water, $D_0 = 8$, $\kappa = 0.5/(\varepsilon_w + D_0)$, $k = (\varepsilon_w - \varepsilon_p) / (D_0 + \varepsilon_p)$ with $\varepsilon_p = 2$ being the dielectric constant of protein.

Note that these fingerprints of the structures are rotation and translation invariant. Furthermore, as a sequence of the amino acids in a macromolecular structure, the protein data bank, RCSB PDB,[34] can be used that is unique. Therefore, the descriptor vector **X** is unique representation of a macromolecule in a dataset. In addition, the descriptor vector **X** has the same length for any set of the macromolecules used as input.

It is important to note that if the chemical sample space of the input descriptor vector becomes quite large, then the so-called *principal components analysis*[35] can be performed to reduce the degrees of freedom.

3 Web-based Services

3.1 Structure

Figure 4 shows a flowchart of the offered web-based services. It consists of several databases using different experimental or quantum mechanics data. In particular, it includes the hydration free energies of molecules database;[12] the heat of formation (or standard enthalpy of formation) of small molecules using quantum mechanics calculations with the Perdew-Burke-Ernzerhof hybrid functional (PBE0).;[5,20] pKa values experimentally calculated for different amino acids in different proteins;[14–17] the experimental changes on the Gibbs free energies of the mutant proteins.[18,19]

The *clients* are the personal computers (PCs), where the users with a login account will be able to access the first layer of the provided web-based services from the main computer, namely the *server*. With that first session login, the users will be allowed to access and manipulate the data from different databases. In particular, the users can read the data from each database in a tabular form, which then can be copied and pasted in the local computer. Besides, the users can see some statistics about the data included in each dataset, allowing for a judgment of the distribution of the data. Also, the users are allowed using the forms provided on the web to upload new data in a particular database. These data will initially be marked as “not checked“, but after the main administrator of the web-based services verifies the authenticity of the information, the data will be added to the existing database. That can allow the databases to increase the amount of information in the future.

Using the data for each dataset, the main administrator, frequently, performs the optimization of the artificial neural network parameter using an exhaustive machine deep-learning approach by employing the BSANN python code, internally in the server. Note that only specific users are also allowed to perform the optimization on the server using special Login information. Then, the second session of login, for specific users, is allowed to use other services of our tools. In particular, those specific users can use web-based services online to design and optimize novel molecules. Then, using the optimized artificial neural network parameters with the training data, they can predict different properties provided for these molecules. It is interesting to note that the external plugins use graphical interfaces, making the services very user-friendly.

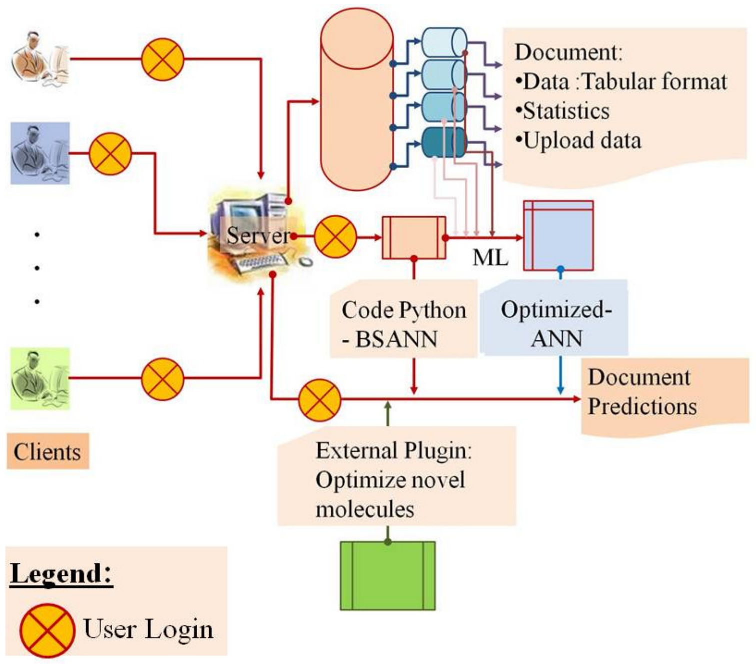


Figure 4: The flowchart of the web based services.

3.2 Datasets

There are four databases served using that web-based service. The first database contains 642 small organic molecules, for which we know the experimental hydration free energies,[12] which has also been subject to our previous studies.[36] The second database contains 1475 experimental pKa values of inozable groups in 192 proteins both wild type (153 proteins) and mutated (39 proteins).[14–17] The third database has 2693 experimental values of the Gibbs free energy changes in 14 mutant proteins.[18,19] The last database has 7101 quantum mechanics heat of formation calculations[5] (and the references therein), the so-called *QM7*, which is a subset of the so-called *GDB13* molecules, optimized at the quantum mechanics level with the Perdew-Burke-Ernzerhof hybrid functional (PBE0).[20]

In Figure 5, we show the distribution of the average experimental pKa for each residue in the wild-type and mutated proteins of the database.

In Figure 6, we show the distribution of the percentage of each type of mutation in the database for which we know either $\Delta\Delta G$ or $\Delta\Delta G^{\mathrm{H_2O}}$, namely 1: single mutation; 2: double

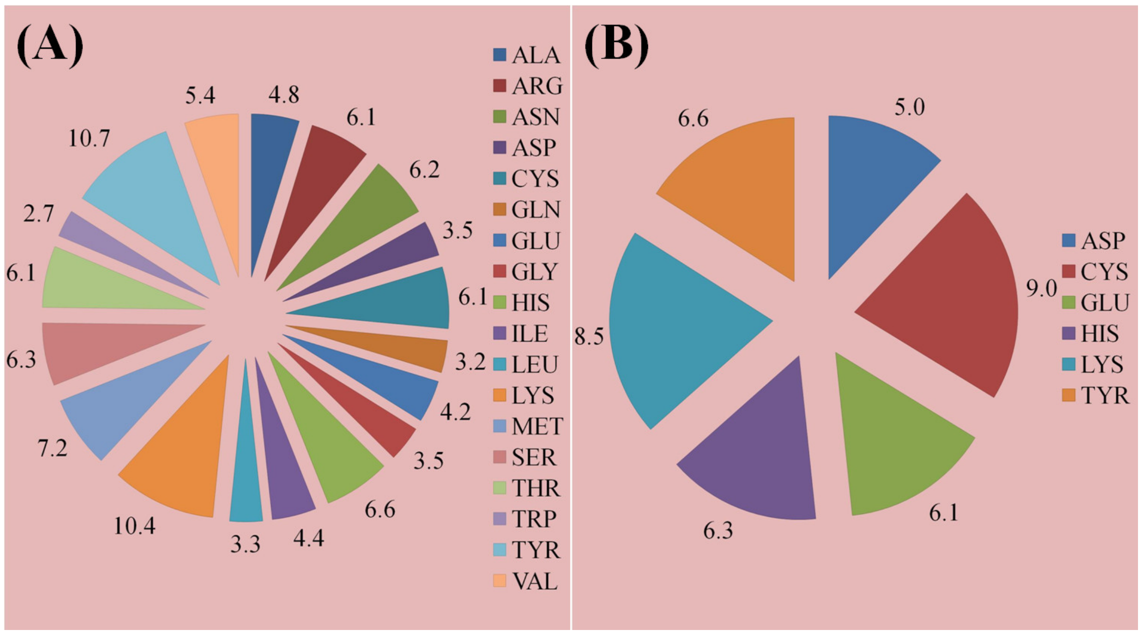


Figure 5: Average experimental pKa for each residue in the wild-type (A) and (B) mutants proteins of the database.

mutations, and so on, up to 6: six mutations.

All the data are prepared and optimized in advance using AMBER force field[21] in CHARMM macromolecular computer simulation program. The BSANN code performing the optimization and prediction is in Python computer programming language. The descriptor vectors of the small molecules are based on the Coulomb matrix and the sum over bonds properties, and for the macromolecular systems they take into account the chemical-physical fingerprints of the region in the vicinity of each amino acid.

Software implementing the methods discussed in this study is free for download from the website `"https://github.com/kamberaj/bsann"`.Also, from the same website, the database of all molecular structure and topology files, used in our calculations prepared using general AMBER force field, can be accessed via the web-based service.

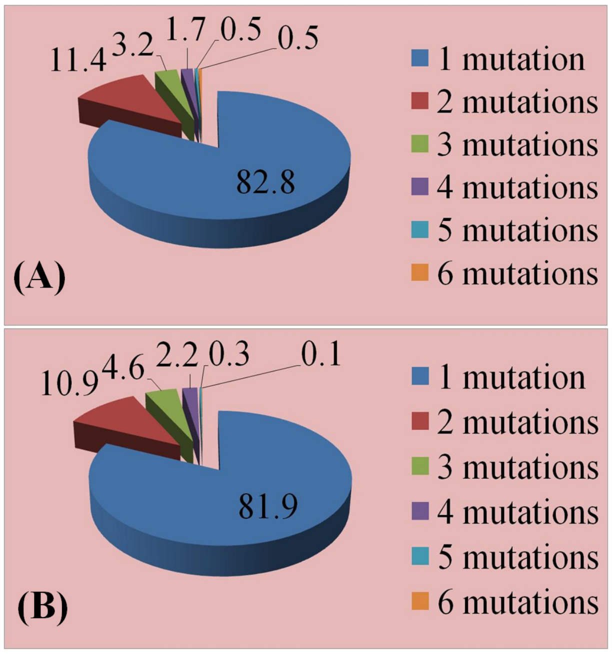


Figure 6: Percentage of each type of mutation in database for which we know either $\Delta\Delta G$ (A) or $\Delta\Delta G^{H_2O}$ (B). 1: single mutation; 2: double mutations, and so on, 6: six mutations.

4 Results

In this section, we show some results of the predictions using different datasets. For that consider a method to learn a function from a finite dataset $\mathcal{D}$ of input-output pairs, namely $(\mathbf{X}, \mathbf{Y})$, where $\mathbf{X}$ is the feature descriptor input vector for each atom and $\mathbf{Y}$ is the reference output vector for each atom, such as the hydration free energy, change on the Gibbs free energy, heat of formation, amino acid pKa, and so on. The dataset is then split into a training dataset $\mathcal{D}_{\text{train}}$ used for learning (or gaining experience) and a validation dataset $\mathbf{D}_{\text{valid}}$ used for testing the knowledge, such that

$$\mathcal{D} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{valid}}$$

In this study, we will discuss the ability of the training dataset to optimize the parameters of the artificial neural network as a function of the size of the training dataset. In particular, we intend to determine optimal average interval of the confidence such that the error in prediction of a value is small; that is, determining the average confidence interval within which we can predict with a certain confidence level (such as 95 %) the value of any new validation data.

It is important to note that "optimal" here does not the necessary mean small value of the average interval. Instead, an optimal average interval is one that provides the highest value of the confidence level, within which most of the predicated values lie in. In the following discussion, we will use the term "match" for such cases, that is, when the prediction interval of error coincides with that provided by the experimental value using statistical confidence of 95 %, which is verified using the so-called *Student t-Distribution*.

To determine the average confidence interval, we used the following statistical justification.[37] Suppose that there are N_{train} training data points, and n is the number of the neural networks defining the bootstrapping model given above. Then, the $100(1-\alpha)$ % bootstrapping confidence interval of the average value for each data point prediction is given by:

$$\left(\bar{Y}_i - c_{i,u}\frac{\sigma_i}{\sqrt{n}}, \bar{Y}_i - c_{i,l}\frac{\sigma_i}{\sqrt{n}}\right) \tag{13}$$

where σ_i is the unbiassed standard deviation obtained from the bootstrapping data distribution. $c_{i,u}$ and $c_{i,l}$ are the upper and lower critical values, respectively, determined from the empirical distribution function F of the bootstrapping dataset as:

$$F(t_i = c_{i,u}) = 1 - \alpha/2$$

$$F(t_i = c_{i,l}) = \alpha/2$$

where t_i is the studentized bootstrapping random variable obtained from the data points of the ith prediction.[37] Then, the average statistical error from all prediction N_{train} data points is calculated using the chain rule as follows:

$$c_u\sigma = \sqrt{\sum_{i=1}^{N_{\text{train}}} (c_{i,u}\sigma_i)^2} \tag{14}$$

$$c_l\sigma = \sqrt{\sum_{i=1}^{N_{\text{train}}} (c_{i,l}\sigma_i)^2} \tag{15}$$

where it is assumed that the statistical errors obtained for each of the training data points are independent, which is indeed the case. Then, the average $100(1-\alpha)$ % bootstrapping confidence interval of the average value for each data point prediction is defined as:

$$\left(\bar{Y}_i - c_u\frac{\sigma}{\sqrt{n}}, \bar{Y}_i - c_l\frac{\sigma}{\sqrt{n}}\right) \tag{16}$$

Eq. 16 is used to determine the upper and lower bound of the average values in all our predictions shown in the graphs of the following discussions. Note that in practice, the Student t-distribution can be used to approximate the distribution of the bootstrapping dataset, and hence $c_l = -c_u = t_{n-1,\alpha/2}$, which is the critical value for the t-distribution. In this study, the confidence level was $\alpha = 0.05$.

4.1 The hydration free energy database

Table 1 summarizes the Pearson coefficient, mean average error (MAE) (in kcal/mol), root mean square error (RMSE) (in kcal/mol) and Matches (in %) for different lengths of training dataset. Predictions are based on the neural network parameters optimized using only the training dataset. Our results show that an MAE value as small as 1.192 kcal/mol is obtained in the validation dataset, corresponding to a Pearson coefficient of 0.886 and an RMSE of 1.770 kcal/mol, for a size of the training dataset of 350 molecules (or equivalently, 84 % of the overall dataset). For that case, the percentage of matches between the predicted and experimental values is 81.5 % with an 95 % statistical confidence.

Table 1: Pearson coefficient, MAE (in kcal/mol), RMSE (in kcal/mol) and Matches (in %) for different numbers of training data. Predictions are based on the neural network parameters optimized using only the training data set.

Pearson	MAE (kcal/mol)	RMSE (kcal/mol)	Matches (%)	Set	Size
0.964	0.539	1.135	97.3	all dataset	415
0.998	0.168	0.231	100.0	training dataset	300
0.883	1.507	2.124	90.4	validation dataset	115
0.970	0.442	1.040	93.7	all dataset	415
0.999	0.170	0.228	97.3	training dataset	330
0.852	1.495	2.254	80.0	validation dataset	85
0.984	0.353	0.742	94.9	all dataset	415
0.998	0.197	0.266	97.4	training dataset	350
0.886	1.192	1.770	81.5	validation dataset	65
0.988	0.272	0.628	92.8	all dataset	415
0.998	0.176	0.258	93.4	training dataset	380
0.878	1.313	1.988	85.7	validation dataset	35

In Figure 8, we present scatter plots of the experimental hydration free energies and predicted free energies for two different training data sizes, $N = 330$ and $N = 380$ molecules. Errors are calculated using the bootstrapping method and the straight line represents the function $f(x) = x$. Besides, we have indicated the 95 % bootstrapping confidence interval of error with parallel lines. Results show that a training dataset of size 380 molecules determines better the confidence interval than the training dataset of size 330 molecules. That is because the distribution of the data points of the training dataset influences on the topology of the input data, and thus, larger the input dataset more insights into the structure of the data distribution can be revealed.

We also used a larger dataset of 630 molecules to train and test the neural network, as shown in Figure 8. For this case, we used 560 (equivalent to 89 % of the total size of the dataset) data points to train the neural network, and the rest about 70 data points for validation (or equivalently, about 11 % of the entire dataset). Our results show an improvement of the predictions when compared with the smaller dataset, as used above; that can be shown by our results presented in Figure 8, indicating that all the validation data points lie inside the 95 % bootstrapping confidence interval. For this dataset, the

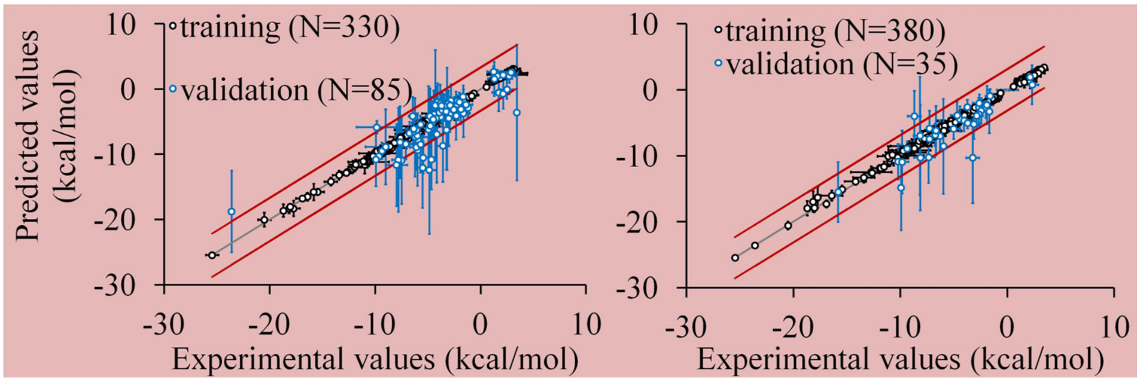


Figure 7: Parity plots between the experimental hydration free energy and predicted free energies. Errors are calculated using the bootstrapping method and the straight line represents the function $f(x) = x$. The dataset size was 415 molecules. The straight red lines represent the boundary of the 95 % bootstrapping confidence interval of error.

following values of MAE, RMSE, and Pearson correlation coefficient R were obtained: For the training data, MAE= 0.208 kcal/mol, RMSE= 0.286 kcal/mol and $R = 0.997$; for the validation data, MAE= 0.732 kcal/mol, RMSE= 1.050 kcal/mol and $R = 0.945$.

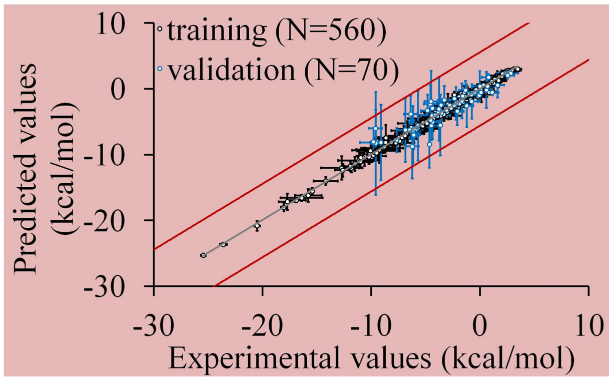


Figure 8: Parity plots between the experimental hydration free energy and predicted free energies. Errors are calculated using the bootstrapping method and the straight line represents the function $f(x) = x$. The dataset size was 630 molecules. The straight red lines represent the boundary of the 95 % bootstrapping confidence interval of error. For the training dataset: MAE= 0.208 kcal/mol, RMSE= 0.286 kcal/mol and $R = 0.997$; for the validation dataset: MAE= 0.732 kcal/mol, RMSE= 1.050 kcal/mol and $R = 0.945$.

4.2 The pKa of amino acids in proteins

Table 2 presents the results of the predictions on both the training and validation datasets. The size of the entire dataset is $N = 953$ pKa calculations. Our results indicate that the Pearson correlation coefficient is above 0.95 in both training and validation dataset. On the training dataset, the smallest MAE and RMSE were 0.104 kcal/mol and 0.164 kcal/mol, respectively. For the validation dataset, the smallest MAE and RMSE values were 0.269 kcal/mol and 0.416 kcal/mol, respectively, obtained for the size of the training dataset about 89 % of the entire dataset. In addition, the matches between the experimental and predicted values of the pKa on the validation dataset is 82 % with a statistical confidence of 95 %.

Table 2: Pearson coefficient, MAE (in kcal/mol), RMSE (in kcal/mol) and Matches (in %) for different numbers of training data. Predictions are based on the neural network parameters optimized using only the training dataset. The sizes of the datasets are $N = 953$ and $N = 1240$ pKa calculations.

		Training data				Validation data			
N_{train}	% of data	Matches %	MAE $\frac{\text{kcal}}{\text{mol}}$	RMSE $\frac{\text{kcal}}{\text{mol}}$	Pearson	Matches %	MAE $\frac{\text{kcal}}{\text{mol}}$	RMSE $\frac{\text{kcal}}{\text{mol}}$	Pearson
					$N = 953$				
750	79	92	0.104	0.164	0.998	74	0.419	0.742	0.951
800	84	91	0.144	0.238	0.996	78	0.310	0.565	0.961
850	89	89	0.118	0.176	0.997	82	0.269	0.416	0.992
					$N = 1240$				
800	65	99	0.034	0.058	1.000	81	0.451	0.843	0.940
1000	81	99	0.037	0.075	1.000	83	0.413	0.738	0.962
1100	89	98	0.038	0.073	1.000	84	0.295	0.485	0.983

In Figure 9, we present the predicated and experimental pKa values graphically as a scatter plot. In addition, we show the average 95 % bootstrapping confidence interval of error of the predicated values within the training dataset. The plots are created for two training data sizes, respectively, 84 % and 89 % of the entire dataset. Interestingly, our results indicate that almost all the validation data predication of pKa values lie inside the average bootstrapping confidence interval of error when 89 % of the dataset is used in training the neural network.

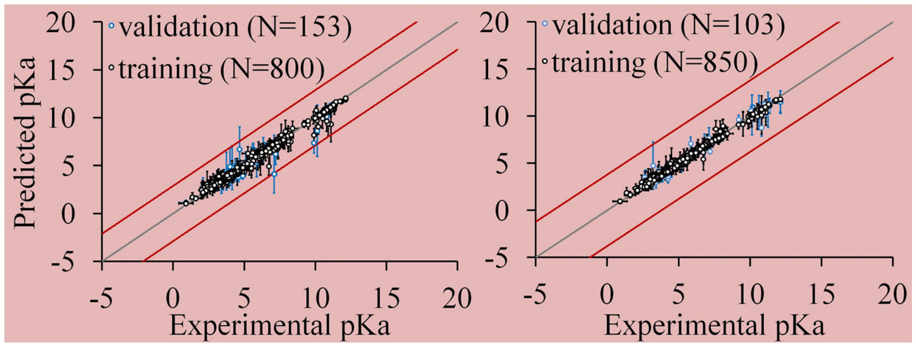


Figure 9: Parity plots between the experimental pKa values and predicted pKa values. Errors are calculated using the bootstrapping method and the straight line represents the function $f(x) = x$. The size of the dataset was $N = 953$. The straight red lines represent the boundary of the 95 % bootstrapping confidence interval of error.

To check the influence of the dataset size on the learning efficiency from the data, we optimized the neural network for a larger dataset of 1240 pKa calculations. We implemented three different training datasets for the optimizations of the neural network to check the influence of the size of the training data set and the length of the entire dataset, which determine the topology of the input data. We notice that the dataset with $N_{\text{train}} = 1100$ (which is about 89 % of the entire dataset) data points for training provided the best optimization. The results are summarized in Table 2, and plotted in Figure 10. Our results show that the MAE decreases about twice for the same percentage of data in the training set taken from a smaller dataset, namely MAE= 0.038 kcal/mol; a smaller RMSE is also obtained for this dataset of about 0.073 kcal/mol. Furthermore, it can be see that the percentage of matches on the training dataset increases to 95 % with a perfect Pearson correlation between the experimental and the predicted of about $R = 1.000$. Moreover, our results show (Figure 9) that the average 95 % bootstrapping confidence interval of error is larger, and all the predicted values of pKa of the validation set lie within bootstrapping confidence interval.

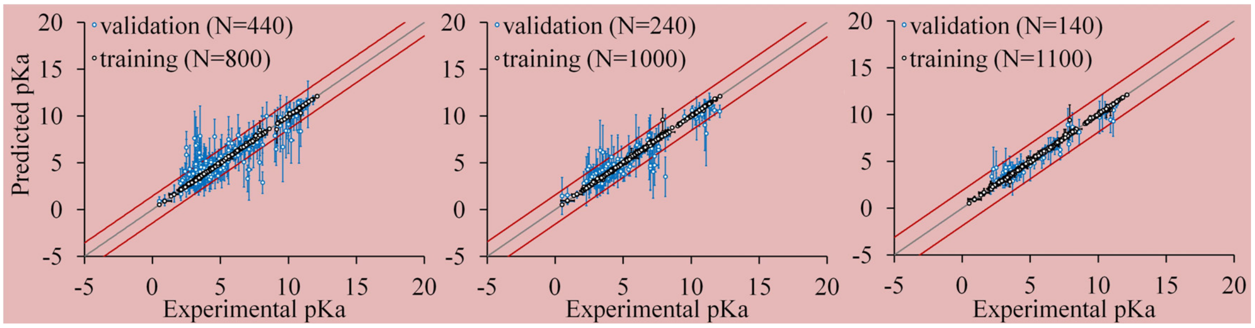


Figure 10: Parity plots between the experimental pKa values and predicted pKa values. Errors are calculated using the bootstrapping method and the straight line represents the function $f(x) = x$. The size of the dataset was $N = 1240$. The straight red lines represent the boundary of the 95 % bootstrapping confidence interval of error.

4.3 Quantum mechanics database

The results of the predicted values of the heat of formation from the quantum mechanics calculations using PBE0 method are shown in Figure 11. The size of the dataset is 7000 molecules. We used two different sets of the training data with lengths 3000 (or equivalently 43 % of the size of the dataset) and 5000 (or equivalently, approximately 71 % of the entire dataset). Our results indicate excellent performance of the predictions using the optimized neural network on the validation data; using just 43 % of the entire dataset for training of the neural network, and the test on the validation data show just a few data are outside the predicted average 95 % bootstrapping confidence interval of error, and when 71 % of the total data are used for training, then all the tested calculations from the validation dataset lie inside the 95 % bootstrapping confidence interval. Interestingly, our results indicate that the optimization of the neural network and hence the increase on the experience for a deep-learning are strongly influenced by the length of the data in the dataset. That explains that the topology of the input dataset plays an important role on the experience gained from the training of the neural network. In the following discussion, we argue that this should be related to the topology of the input data.

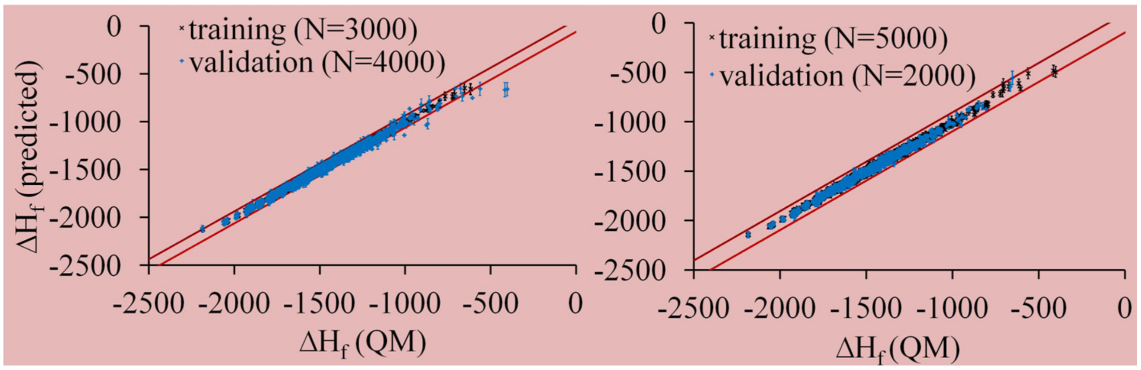


Figure 11: Parity plots between the quantum mechanics heat of formation values and predicted values. Errors are calculated using the bootstrapping method and the straight lines represent the boundary of the 95 % bootstrapping confidence interval of error. Quantum mechanics heat of formation is calculated using the PBE0 method.

5 Discussion

In this work, we intend to establish a methodology for an automated machine-like deep learning approach for predicting different (macro)molecular properties. In particular, for a training dataset of molecules, $\mathcal{D}$ created of N_{train} pairs (X_i, Y_i) for $i = 1, 2, \cdots, N_{\text{train}}$, where the vector $\mathbf{X}$ denote the feature descriptor vector of dimension $N_{\text{features}} \times N_{\text{train}}$ and $\mathbf{Y}$ of dimensions $N_{\text{properties}} \times N_{\text{train}}$ the reference values. That aims to obtain an estimate of the probability $P(\mathbf{Y}^\star|\mathcal{D}, \mathbf{X}^\star)$ to predict the output $\mathbf{Y}^\star = \left(Y_1^\star, Y_2^\star, \cdots, Y_{\text{properties}}^\star\right)$ of an optimized neural network for any input test data-point $\mathbf{X}^\star = (X_1^\star, X_2^\star, \cdots, X_{\text{features}}^\star)$. This calculation is now an automated process since the black box is trained to predict the output value described by the probability $P(\mathbf{Y}^\star|\mathcal{D}, \mathbf{X}^\star)$, which makes the parameterization of the force fields a very efficient automation process.

However, the accuracy in estimation of $P(\mathbf{Y}^\star|\mathcal{D}, \mathbf{X}^\star)$ is a data-driven process, and the prediction of $\mathbf{Y}^\star$ depends on the used training dataset. In particular, it depends on the diversity of the feature descriptors for the dataset of molecules, that is, the amount of N_{features}. Besides, it depends on the size of the dataset, N_{train}. Both the diversity of the feature descriptors of the compounds and the size of the dataset are interconnected; however, a large size dataset is practically difficult to be established due to the lack of the experimental data, and quantum mechanics data may be expensive to obtain. Besides, increasing the dimensions of the input feature descriptor vector, $N_{\text{features}} \times N_{\text{train}}$, is equivalent to increasing the amount of information processed by the computer, and hence it increases the amount of the irreversible heat generated during the processing.[38] That is related to another computer term, namely "big-data" processing. In the following discussion, we will try to quantify the *weight* of the big-data information by using physical interpretation of information and the principle of the equivalence mass-energy-information.[39,40] That will allow us to establish an equivalence between the (necessary) amount of the input training data for accurate prediction of $P(\mathbf{Y}^\star|\mathcal{D}, \mathbf{X}^\star)$ and the limit of the amount of the information that can be processed by a computer considering the heat generated during the computer processing, and so the amount of the external work necessary to process that big-data of information by a computer.

Mass **of information**

For that, consider that the input feature descriptor vectors are represented in binary form, that is, as zeros and ones. Then, the size of the input matrix of features for all training dataset of compounds is:

$$B = N_{\text{train}} N_{\text{features}} \text{ (bits)} \tag{17}$$

Based on the Landauer principle,[38,39] process of adding information into the storage device requires some work done by an external agent:

$$W \geq k_B T \ln 2 \tag{18}$$

where k_B is the Boltzmann's constant ($k_B = 1.38064 \times 10^{-23}$ J/K) and T is the temperature of the environment in kelvin. That work is necessary to change the physical conditions of the environment to add one bit of information into the storage device, which equals the amount of heat generated for creating one but of information in the storage device, and hence:

$$\Delta h_{\text{bit}} \geq k_B T \ln 2 \tag{19}$$

Note that has already been measured experimentally (see, for example, Ref.[40] and the references therein). Then, the minimum total amount of the heat generated for storing B bits in a storage device at $T = 300$ K is

$$\Delta H_{\text{bits}} = (k_B T \ln 2)\, B \approx 0.018 B \text{ eV} \tag{20}$$

knowing that 1 eV $= 1.602 \times 10^{-19}$ J.

For example, suppose that we have 10^4 training data points and $N_{\text{features}} = 100$ for each compound (see also Figure 12), then the total minimum amount of heat generated is:

$$\Delta H_{\text{bits}} = 0.018 \times 10^6 \text{ eV} = 18 \text{ keV} \tag{21}$$

Furthermore, the minimum amount of heat generated for adding one feature (or equivalently, one single bit of information) to each compound of the training dataset is

$$\Delta H_{\text{bits}} = 0.018 \times 10^4 \text{ J} = 0.18 \text{ keV} \tag{22}$$

Using the Einstein principle of the mass-energy equivalence, the rest energy of a particle with mass m is given

$$E = mc^2 \tag{23}$$

where c denotes the speed of light in vacuum, $c = 3.00 \times 10^8$ m/s. Eq. 23 indicates a well-known fact that particles (such as photons) with zero rest mass have zero rest energy. In analogy with mass-energy principle of special theory of relativity, another principles has been postulated of the mass-energy-information equivalence:[40]

$$\Delta h_{\text{bit}} = \gamma m_{\text{bit}} c^2 \tag{24}$$

where m_{bit} denotes the mass of one bit of information in SI units of kilogram, and γ is

$$\gamma = \frac{1}{\sqrt{1 - \frac{v^2}{c^2}}}$$

where v would represent the speed of storing the information.

Therefore, we have

$$\gamma m_{\text{bit}} = \frac{k_B T \ln 2}{c^2} \tag{25}$$

That indicates that when $T = 0$ K (absolute zero temperature), then $m_{\text{bit}} = 0$ kg. We can assume that absolute zero temperature corresponds to the case when there is no information stored in the device; therefore, for an empty digital storage device $m_{\text{bit}} = 0$, and then every added bit of information increases the mass by[40]

$$\gamma \Delta m_{\text{bit}} = \frac{k_B T \ln 2}{c^2} \tag{26}$$

For example, at room temperature of the environment, $T = 300$ K, we have

$$\gamma \Delta m_{\text{bit}} c^2 = 0.018 \text{ eV}$$

or

$$\gamma \Delta m_{\text{bit}} \approx 3.19 \times 10^{-38} \text{ kg}$$

If we consider only the energy of the bit stored in the digital device at rest, then $\gamma = 1$, and

$$\Delta m_{\text{bit}} \approx 3.19 \times 10^{-38} \text{ kg}$$

That could give an indication of how fast the bit information moves, the speed of the storing the information in the digital storage device. If we consider again the case of the $10^4 \times 10^2 = 10^6$ bits of the training data, then the total increase on the mass of the digital device is:

$$\Delta M_{\text{bits}} = 18 \text{ keV}$$

Besides, the increase of the mass of information for adding one bit of information to each input feature descriptor vector of the 10^4 training data is

$$\Delta M_{\text{bits}} = 0.18 \text{ keV}$$

It is interesting to emphasize how beneficial could be if instead of cooling systems to invent a method of how to reuse all that energy released in the form of the irreversible heat.

In addition, one should also consider the energy generated in the form of the irreversible heat by computer processing of that data, as shown in Figure 12. Similar analysis can be followed in that case.

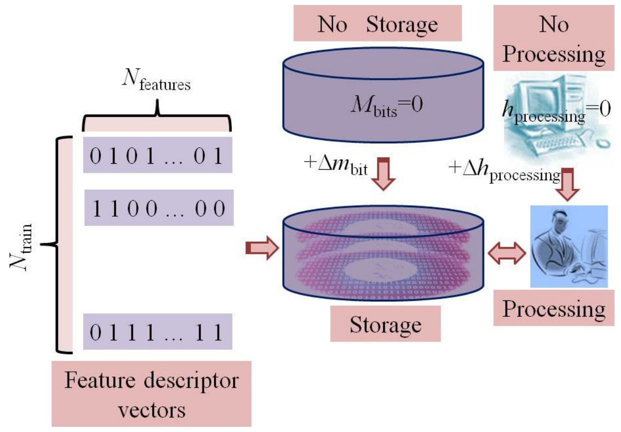


Figure 12: A diagram of the storage and computer processing of the input information. M_{bits} gives the mass in kilogram of the amount of information stored in device in bits based on the mass-energy-information equivalence principle;[40] Δm_{bit} denotes the increase of the mass by adding one bit to the storage device; $\Delta h_{\text{processing}}$ gives the amount of the (irreversible) heat released during the computer processing of the information.[39]

Note that we can also take advantages of new advanced computer architectures to develop new more efficient algorithms. In particular, quantum architecture of the quantum computing can be useful in developing the so-called *quantum artificial neural networks.*[41–43] In analogy with quantum computing, the quantum bit, the so-called the *qubit*, can be introduced:

$$\mid \phi\rangle = \alpha \mid 0\rangle + \beta \mid 1\rangle$$

where $\mid 0\rangle$ and $\mid 1\rangle$ are the basis vectors, given as

$$\mid 0\rangle = (1,\ 0)^T, \quad \mid 1\rangle = (0,\ 1)^T$$

and α and β are complex numbers such that $\mid \alpha \mid^2$ and $\mid \beta \mid^2$ give the probability of measuring the state $\mid 0\rangle$ and $\mid 1\rangle$, respectively, and hence $\mid \alpha \mid^2 + \mid \beta \mid^2 = 1$. Based on the quantum artificial neural network model, every input vector can be represented by a qubit, namely $\mid X_i\rangle$, for $i = 1,\ 2,\ \cdots,\ N_{\text{train}}$. The output of some layer j is then

$$\mid Y_j\rangle = \hat{F}\left(\sum_{i=1}^{N} \hat{W}_{ij} \mid X_i\rangle + \mid b_j\rangle\right)$$

where N is the number of neurons, $\hat{F}$ is the quantum operator activation function, $\hat{W}_{ji}$ is a $2^n \times 2^n$ matrix operator with n being the number of qubits acting on the input $\mid X_i\rangle$ (where $n = 1$ in that case, as suggested elsewhere[44]). Here, $\mid b_j\rangle$ qubit denotes the bias term. A learning rule has also been proposed to update the matrix elements $\hat{W}_{ij}$ to the desired output qubit $\mid o_j\rangle$ as follows:

$$\begin{aligned} \hat{W}_{ij}(t+1) &= \hat{W}_{ij}(t) + \gamma_w \left(\mid o_j\rangle - \mid Y_j(t)\rangle\right) \langle X_i \mid \\ \mid \hat{b}_j(t+1)\rangle &= \mid \hat{b}_j(t)\rangle + \gamma_w \left(\mid o_j\rangle - \mid Y_j(t)\rangle\right) \end{aligned} \tag{27}$$

where γ_w is the usual learning constant rate, a number between 0 and 1, and t is the iteration step. A comparison of different proposed quantum artificial neural network models is presented in Ref.[44]

Based on the above formalism, then every qubit representing the feature descriptor vectors will increase the mass of the information for 10^4 training data by

$$\Delta M_{\text{qubit}} = 0.18 \text{ keV}$$

which is 100 times smaller than the classical artificial neural network.

Topological Data Analysis

It has been argued elsewhere[24] that the diversity of the feature descriptors of the compound database is essential to increase the range of the test data that can be predicted since the machine learning methodology works very well in interpolating the new data points, but suffers on extrapolating new data outside the range covered by the training dataset. Therefore, one of the critical future developments of the automated machine learning methodologies is the choice of the training dataset and the feature descriptors of the chemical compounds. In Ref.,[24] a topological data analysis tools is discussed to analyze the feature descriptors of the molecules, which will be introduced in the following. The topological data analysis (TDA) is a field dealing with the topology of the data to understand and analyze large and complex datasets.[45,46] Here, we are analyzing the dataset represented by a vector of feature descriptors of length N and each data point has a dimension D:

$$\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}, \quad \mathbf{x}_i = \{x_{i1}, x_{i2}, \cdots, x_{iD}\} \tag{28}$$

For example, N may represent the number of molecules in the dataset and D number of specific features for each molecule. Moreover, as in Ref.,[24] we assume that the data of the dataset are hidden in a "black box", for example, a *database*, and also, they are about to be used by a machine learning, which is another "black box". In such a situation, knowing about the topology of the data (*e.g.*, the sparsity of the data points) is of great interest. Note that the TDA is applicable even when the user has access to the data, that is, the structure of the molecules of the dataset is known a priory. In such a situation, the TDA can be applied to determine the topology of the key feature descriptors for each molecule. Note that the TDA is employed to reveal the intrinsic persistent features of the DNA and RNA.[47,48] Therefore, the construction of the topological spaces upon the input data of a machine learning approach can be applied for each dimension separately, namely to the time series of the form $\mathbf{X}_d = \{x_{1d},\, x_{2d},\, \cdots,\, x_{Nd}\}$, or for each molecular structure, namely $\mathbf{X}_k = \{x_{k1},\, x_{k2},\, \cdots,\, x_{kD}\}$. But, it can also apply to both dimensions at the same time, for instance, by constructing the input data in the form of the following time series obtained by aligning feature descriptors of the molecular structures in one dimension:

$$\mathbf{X} = \{x_{11},\, \cdots,\, x_{1D}, x_{21},\, \cdots,\, x_{2D}, \cdots, x_{N1},\, \cdots,\, x_{ND}\} \tag{29}$$

In that case, the input vector of the feature descriptors is a time series of length $N_{\text{train}} = ND$.

Then, to determine the topological space for this dataset, we first define a distance $\sigma > 0$. The Vietoris-Rips simplicial complex $R(\mathbf{X},\, \sigma)$ or simply Rips complex for each $k = 1,\, 2,\, \cdots$ as a k-simplex of vertices $\mathbf{X}_i^k = \{x_{i_1},\, x_{i_2},\, \cdots,\, x_{i_k}\}$ such that they satisfy the condition that the mutual distances between any pair of the vertices is less than σ:

$$d(x_{i_k},\, x_{i_l}) \leq \sigma, \quad \forall x_{i_k},\, x_{i_l} \in \mathbf{X}_i^k \tag{30}$$

With other words, a k-simplex is part of a $R(\mathbf{X}, \sigma)$ for every set of k data points that are distinct from each-other at a resolution σ and hence the Rips complexes form a filtration of the data from the dataset at a resolution σ. That is, for any two values of the resolution σ and σ' such that $\sigma < \sigma'$, then

$$R(\mathbf{X}, \sigma) \subseteq R(\mathbf{X}, \sigma')$$

where $\subseteq$ denotes the subset.

All the vertices of a k-simplex can be connected in a two-dimensional space by undirected edges forming a graph, which can have different two-dimensional shapes. Figure 13 illustrates how to build simplicial complexes using a set of point cloud data by increasing the resolution value σ.

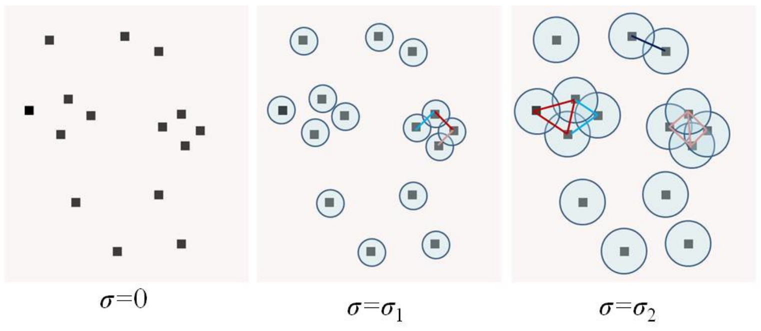


Figure 13: An illustration of building the simplicial complexes by increasing the resolution value σ.[24]

The k-simplex dataset points form a loop that is called *hole*. By increasing the resolution σ, the shapes grow, and some of the holes die, and some new holes are born. This process is the so-called σ loop expansion. The interval between birth and death of a hole is called *persistence interval* indicating whether a hole is structurally relevant or just a noise into the data.

Persistent homology (PH) is an essential tool of TDA, which aims to construct a topological space gradually upon the input dataset, which is done by growing shapes based on the input data. Persistent homology measures in this way the persistence interval of the topological space. The features will be identified as persistent if after the last iteration they are still present.

This procedure is analog to systematic coarse-graining and is of crucial importance for any attempt at capturing natural feature descriptors in terms of a few relevant degrees of freedom, and thus they form the essential philosophical basis of a dataset for the machine learning approaches.

We can argue that the fundamentals of the PH notions on the relevance or irrelevance of perturbations in the data analysis are crucial, and the persistence homology can be considered as necessary as the renormalization group theory in statistical physics when applied to equilibrium phenomena in understanding the relevant or irrelevant interactions. In this analogy, the resolution scaling σ on the topological data analysis can be considered similar to the characteristic correlation length scale that determines the judgment of the strong interactions and correlations renormalization group theory.[49]

Feature descriptor vectors in higher dimensions

Note that in our discussion above, the feature descriptor vectors for each compound are considered time invariant, and thus they represent only two- and three-dimensional feature descriptors of the (macro)molecules. However, higher feature descriptor vectors can also be constructed, such as four-dimensional feature descriptor vectors by including the time as the fourth dimension. In that case, to construct the three-dimension part of the feature descriptor vectors, different conformations of the compounds can be taken into considerations, for example, as generated from the molecular dynamics simulations. In that case, the three-dimensional configurations of the compounds generated from the simulations can be mapped in a three-dimensional grid, where the centers of the grid points will represent the average positions of each atom obtained from its fluctuations after the configurations are aligned to remove the overall translation and rotation motion of the compounds. Therefore, the feature descriptor vectors obtained from these average structures mapped in a three-dimension grid are translation and rotation invariant. A review of such higher dimensional descriptor vectors is discussed in Ref.[50]

6 Conclusions

In this study, we presented a web-based service for automation of (macro)molecular properties predictions using a new algorithm integrated into a machine learning approach. That web-service is made up of four different databases of both molecular and macromolecular systems properties. Each database has a user- friendly interface that provides the possibility to upload information into the database, which then is verified by the main administrator of the service. Besides, the clients can perform statistics on the web related to each database, obtaining in this way the information contained in each database in a tabular or graph format.

Besides, our web-based service provides other tools and plugins for prediction of the properties of the new (macro)molecular systems using a newly developed deep-learning approach based on the bootstrapping swarm artificial neural network. Furthermore, we showed, in this study, how to create input descriptor vector for the artificial neural network for both small molecules and macromolecular systems. The descriptor features included both the two-dimensional (macro)molecular fingerprints and the three-dimensional structure of the systems. Moreover, we presented a statistical approach of how to estimate the bootstrapping confidence interval of the error.

The application of that new algorithm on our data indicated that the topological spaces of molecular properties description vector on the relevance or irrelevance of perturbations in the data analysis are crucial. Furthermore, we envision that the persistence homology can be considered as necessary as the renormalization group theory in statistical physics when applied to equilibrium phenomena in understanding the relevant or irrelevant interactions. In this analogy, the resolution scaling factor on the topological data analysis can be considered similar to the characteristic correlation length scale that determines the judgment of the strong interactions and correlations renormalization group theory.

ASSOCIATED CONTENT

Supporting Information Available

The Web-based Services along with the databases are published on the following website.

Acknowledgement

The author (H.K.) thanks the International Balkan University for the support.

References

(1) Mater, A. C.; Coote, M. L. Deep-learning in chemistry. *J. Chem. Inf. Model.* **2019**, *59*, 2545–2559.

(2) Lubbers, N.; Smith, J. S.; Barros, K. Hierarchical modelling of molecular energies using a deep neural network. *J. Chem. Phys.* **2018**, *148*, 241715–8.

(3) Gastegger, M.; Schwiedrzik, L.; Bittermann, M.; Berzsenyi, F.; Marquetand, P. wACSF-Weighted atom-centered symmetry functions as descriptors in machine learning potentials. *J. Chem. Phys.* **2018**, *148*, 241709–11.

(4) Goh, G. B.; Siegel, C.; Vishnu, A.; Hodas, N.; Baker, N. How Much Chemistry Does a Deep Neural Network Need to Know to Make Accurate Predictions? 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). 2018; pp 1340–1349.

(5) Collins, C. R.; Gordon, G. J.; von Lilienfeld, O. A.; Yaron, D. J. Constant size descriptors for accurate machine learning models of molecular properties. *J. Chem. Phys.* **2018**, *148*, 241718–11.

(6) Schneider, E.; Dai, L.; Topper, R. Q.; Drechsel-Grau, C.; Tuckerman, M. E. Stochastic Neural Network Approach for Learning High-Dimensional Free Energy Surfaces. *Phys. Rev. Lett.* **2017**, *119*, 150601.

(7) Kamath, A.; Vargas-Hernández, R. A.; Krems, R. V.; Carrington, T. J.; Manzhos, S. Neural networks vs Gaussian process regression for representing potential energy surface: A comparative study of fit quality and vibrational spectrum accuracy. *J. Chem. Phys.* **2018**, *148*, 241702–7.

(8) Herr, J. E.; Yao, K.; McIntyre, R.; Toth, D. W.; Parkhill, J. Metadynamics for training neural network model chemistries: A competitive assessment. *J. Chem. Phys.* **2018**, *148*, 241710–9.

(9) Wehmeyer, C.; Noé, F. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* **2018**, *148*, 241703–9.

(10) Chen, R.; Liu, X.; Jin, S.; Lin, J.; Liu, J. Machine learning for drug-target interaction prediction. *Molecules* **2018**, *23*, 2208–15.

(11) Decherchi, S.; Berteotti, A.; Bottegoni, G.; Rocchia, W.; Cavalli, A. The ligand binding mechanism to purine nucleoside phosphorylase elucidated via molecular dynamics and machine learning. *Nat. Communic.* **2015**, *6:6155*, 1–10.

(12) Mobley, D. L. Experimental and Calculated Small Molecule Hydration Free Energies. *http://www.escholarship.org/uc/item/6sd403pz* **Accessed date July 19, 2015**, *UC Irvine: Department of Pharmaceutical Sciences, UCI*.

(13) Riquelme, M.; Lara, A.; Mobley, D. L.; Verstraelen, T.; Matamala, A. R.; V.-Martinez, E. Hydration free energies in the FreeSolv dtabase calculated with polarized iterative Hirshfeld charges. *J. Chem. Inf. Model.* **2018**, *58*, 1779–1797.

(14) Thurlkill, R. L.; Grimsley, G. R.; Scholtz, J. M.; Pace, C. N. pK values of the ionizable groups of proteins. *Protein Science* **2006**, *15*, 1214–1218.

(15) Pace, C. N.; Grimsley, G. R.; Scholtz, J. M. Protein ionizable groups: pK values and their contribution to protein stability and solubility. *J. Biol. Chem.* **2009**, *284*, 13285–13289.

(16) Click, T. H.; Kaminski, G. A. Reproducing basic pKa values for turkey ovomucoid third domain using a polarizable force field. *J. Phys. Chem. B* **2009**, *113*, 7844–7850.

(17) Pahari, S.; Sun, L.; Alexov, E. PKAD: a database of experimentally measured pKa values of ionizable groups in proteins. *Database* **2019**, 1–7.

(18) Gromiha, M. M.; An, J.; Kono, H.; Oobatake, M.; Uedaira, H.; Kitajima, K.; Sarai, A.

ProTherm: thermodynamic database for protein and mutants. *Nucleic Acids Research* **1999**, *27*, 286–288.

(19) Bava, K. A.; Gromiha, M. M.; Uedaira, H.; Kitajima, K.; Sara, A. ProTherm, version 4.0: thermodynamic database for protein and mutants. *Nucleic Acids Research* **2004**, *32*, D120–D121.

(20) Rupp, M.; Tkatchenko, A.; Müller, K. R.; von Lilienfeld, O. A. Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning. *Phys. Rev. Lett.* **2012**, *108*, 058301.

(21) Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. Development and testing of a general amber force field. *Journal of Computational Chemistry* **2004**, *25*, 1157–1174.

(22) Brooks, B. R. et al. CHARMM: The bimolecular simulation program. *J. Comput. Chem.* **2009**, *30*, 1545–1614.

(23) McCulloch, W. S.; Pitts, W. H. A logical calculus of the ideas immanent in neural nets. *Bull. Math. Biophys.* **1943**,

(24) Kamberaj, H. *Molecular Dynamics Simulations in Statistical Physics. Theory and Applications*; Springer-Verlag, 2019.

(25) Qian, N. On the momentum term in gradient descent learning algorithms. *Neural Networks* **1999**, *12*, 145–151.

(26) Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. *J. Mach. Learn. Res.* **2014**, *15*, 1929.

(27) Paass, G. In *Advances in Neural Information Processing Systems 5*; Hanson, S. J., Cowan, J. D., Giles, C. L., Eds.; Morgan-Kaufmann, 1993; pp 196–203.

(28) Zhou, Z.; Kearnes, S.; Li, L.; Zare, R. N.; Riley, P. Optimization of molecules via deep-reinforcement learning. *Scientific Reports* **2019**, *9*, 10752.

(29) Weininger, D. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36.

(30) Unke, O. T.; Meuwly, M. A reactive, scalable, and transferable model for molecular energies from a neural network approach based on local information. *J. Chem. Phys.* **2018**, *148*, 241708–15.

(31) Yang, K.; Swanson, K.; Jin, W.; Coley, C.; Eiden, P.; Gao, H.; G.-Perez, A.; Hopper, T.; Kelley, B.; Mathea, M.; Palmer, A.; Settels, V.; anf K. Jensen, T. J.; Barzilay, R. Analyzing Learned Molecular Representations for Property Prediction. *J. Chem. Inf. Model.* **2019**, *59*, 3370–3388.

(32) Mehler, E. L. In *Molecular Electrostatic Potential: Concepts and Applications*; Murray, J. S., Sen, K., Eds.; Elsevier Science: Amsterdam, 1996; Vol. 3; pp 371–405.

(33) Mehler, E. L.; Guarnieri, F. A self-consistent, micro-environment modulated screened Coulomb potential approximation to calculate pH-dependent electrostatic effects in proteins. *Biophys. J.* **1999**, *77*, 3–22.

(34) Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. *The Protein Data Bank Nucleic Acids Research* **2000**, *28*, 235–242.

(35) Karhunen, K. Über Lineare Methoden in der Wahrscheinlichkeitsrechnung. *Ann. Acad. Sci. Fenn. A1* **1947**, *37*, 1–79.

(36) Izairi, R.; Kamberaj, H. Comparison Study of Polar and Nonpolar Contributions to Solvation Free Energy. *J. Chem. Inf. Model.* **2017**, *57*, 2539–2553.

(37) Dekking, E. M.; Kraaikamp, C.; Lopuhaä, H. P.; Meester, L. E. *A modern introduction to probability and statistics. Understanding why and how*; Springer-Verlag: London, 2005.

(38) Landauer, R. Irreversibility and heat generation in the computing process. *IBM Journal of Researchand Development* **1961**, *5*, 183–191.

(39) Landauer, R. The physical nature of information. *Phys. Rev. A* **1996**, *217*, 188–193.

(40) Vopson, M. M. The mass-energy-information equivalence principle. *AIP Advances* **2019**, *9*, 095206–4.

(41) Lewenstein, M. Quantum perceptrons. *Journal of Modern Optics* **1994**, *41*, 2491–2501.

(42) Zhou, R.; Ding, Q. Quantum M-P neural network. *International Journal of Theoretical Physics* **2007**, *46*, 3209–3215.

(43) Panella, M.; Martinelli, G. Neural networks with quantum architecture and quantum learning. *International Journal of Circuit Theory and Applications* **2011**, *39*, 61–77.

(44) de P. Neto, F. M.; da Silva, A. J.; Ludermir, T. B.; de Oliveira, W. R. Analysis of Quantum Neural Models. XIII Brazilian Congress on Computational Intelligence. 2017.

(45) Carlsson, G. Topology and data. *Bull. Amer. Math. Soc.* **2009**, *46*, 255.

(46) Edelsbrunner, H.; Harer, J. *Computational Topology: An introduction*; Amer. Math. Soc., 2010.

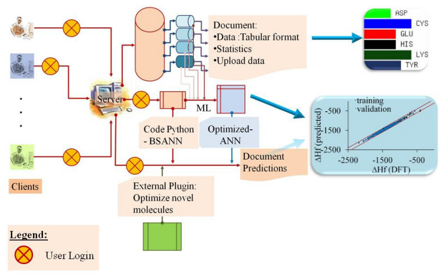
(47) Xia, K.; Zhao, Z.; Wei, G. W. Multiresolution Persistent Homology for Excessively Large Biomolecular Datasets. *J. Chem. Phys.* **2015**, *143*, 134103.

(48) Mamuye, A. L.; Rucco, M.; Tesei, L.; Merelli, E. Persistent homology analysis of RNA. *Mol. Based Math. Biol.* **2016**, *4*, 14–25.

(49) Täuber, U. C. Renormalization Group: Applications in Statistical Physics. *Nuclear Physics B Proceedings Supplement* **2011**, *00*, 1–28.

(50) Peter, S. C.; Dhanjal, J. K.; Malik, V.; Radhakrishnan, N.; Jayakanthan, M.; Sundar, D. In *Encyclopedia of Bioinformatics and Computational Biology*; Ranganathan, S., Gribskov, M., Nakai, K., Schönbach, C., Eds.; Academic Press: Oxford, 2019; pp 661–676.

Graphical TOC Entry


Server
Document:
•Data :Tabular format
•Statistics
•Upload data
ASP
CYS
GLU
HIS
LYS
TYR
ML
Code Python - BSANN
Optimized-ANN
training
validation
ΔHf (predicted)
-500
-1500
-2500
ΔHf (DFT)
Document Predictions
Clients
External Plugin: Optimize novel molecules
Legend:
User Login