Representation of Potential Energy Surfaces using Neural Networks

1Umme Kulsum, 2Raza Imam, 3Mohd Abdullah Khan, 4Asra Ansari
1Student, 2Student, 3Student, 4Student
1Department of Chemistry, Aligarh Muslim University, India,
2Department of Computer Science, Aligarh Muslim University, India,
3Department of Computer & Engineering, Jamia Hamdard University, India,
4Department of Electronics Engineering, Aligarh Muslim University, India

Abstract **- Deep learning is ideally suited for modelling nonlinear potential-energy surfaces, expressing quantum-mechanical interactions, and expanding chemical compound space research. Given the presence of hidden layers, neural networks do more effective predictive analyses as the neural network employs the multiple hidden layers to improve prediction accuracy. There is a requirement for precise potentials that can swiftly repeat high-quality results since the interactions in force fields are represented by a variety of different functions. In this work, we strive to investigate the representation of Potential Energy Surfaces, a crucial component of chemical dynamics, using neural networks. We developed neural network models that can be applied widely to fit one-dimensional data and two-dimensional potential energy surfaces separately. Our methodology concludes different key analytical outcomes as well as crucial future directions that aim to strengthen the potential of chemical dynamics and machine learning.**

keywords **- Potential Energy Surfaces, Neural Networks, Morse Potential, Activation Function, Dimensional Curves**

I. INTRODUCTION

Chemical dynamics include the Potential Energy Surface PES. It demonstrates how the (relative) position of the atom affects the potential energy of a chemical system.[1] The Lennard- Jonnes (LJ) and Morse potentials (MP) are the two traditional pair potentials utilized in modelling. Each has two movable parameters and comprises of a short-range repulsion and a long-range attraction. The Morse potential energy function has the following formula: V(r) = D(1−e−a(r−re))2. Here, re denotes the equilibrium bond distance, while r denotes the distance between the atoms, D is the optimal depth (determined in relation to the dissociated atoms), and a determines the potential's width (the smaller the value of a the larger the well). Because it clearly accounts for the consequences of bond breakage, such as the existence of unbound states, it provides a more accurate representation of the vibrational structure of the molecule than the quantum harmonic oscillator. Additionally, it considers the non-zero transition probability for overtone and combination bands as well as the anharmonicity of real bonds.

A straightforward extension of Morse potential energy function (See Figure 1(a)) comes from adding a second configuration-space dimension, referred to as the 2-dimensional Morse potential. Similarly, the Lennard-Jonnes (LJ) potential (See Figure 1(b)) has the form V = E[(/r)−12 - (σ/r)−6] . The (σ/r)−6 cohesion is based on Van der Waals interactions, while the e−ar is motivated by a screened Coulomb potential. The repulsive terms were invented Ad Hoc [2].

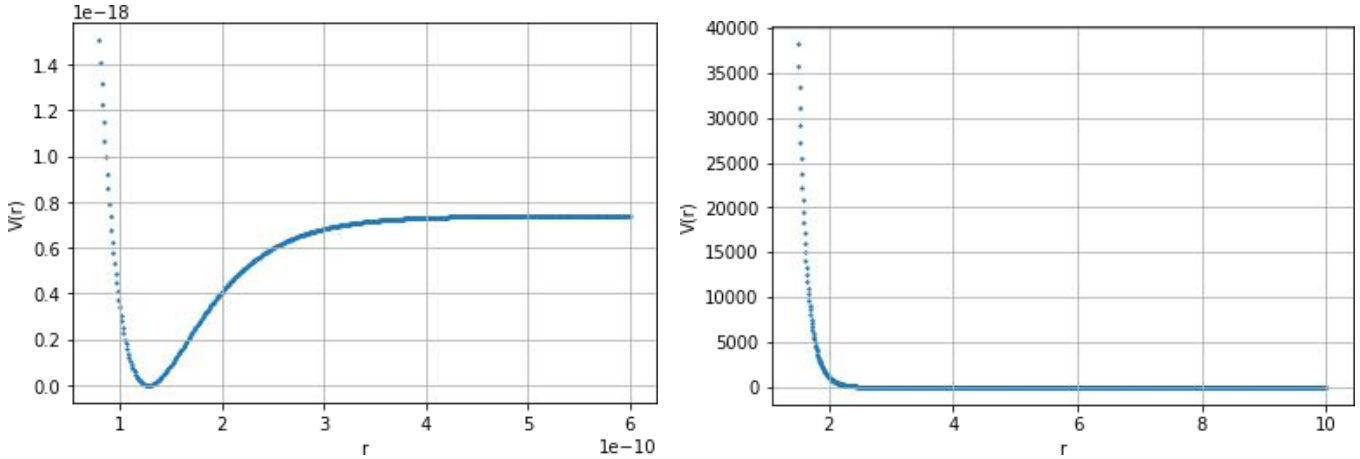

Figure 1. (a) Morse potential with D = 7.10e−19J and β = 1.8e+10, and **(b)** Lennard Jonnes potential with E = 3.180 meV and σ = 2.928˚A

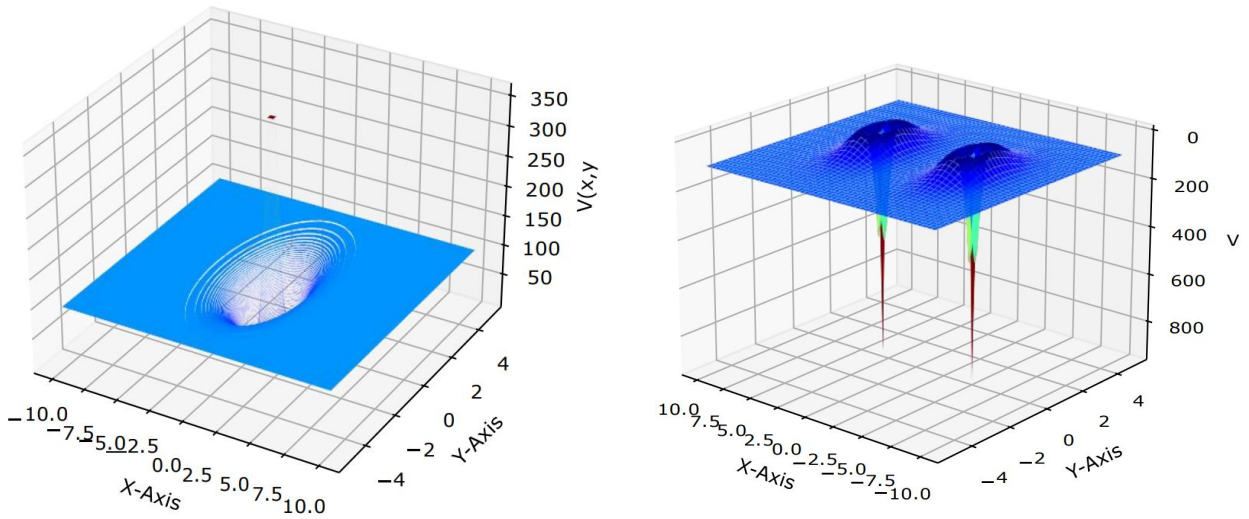

Figure 2. (a) 2D Morse potential with parameter values De = 100, k = 200, re = 1, and **(b)** Double 2D Morse potential with parameter values De = 100, k = 200, re = 1. The upper panel has b = 3, and the lower panel has b = 5

The fitting of PES has come up as an essential application of machine learning. Curve fitting essentially involves finding a function G(q) which would help us in finding the value of our function F(q) at an unknown q. To fit the data to this analytical function, we use the linear least square method. In the linear least squares method, we aim to minimize the sum of the squares of the errors with respect to a set of chosen parameters [3].

Using a Neural Network (NN) to fit curves gives us a better way to describe complex potentials and allows us to take those potentials into many dimensions. The operation of a biological neuron serves as a model for neural networks. Like neurons, the neural networks transmit information from one node to another, and the output is then received based on the input information's weighted value and the transfer function in use. A computing unit known as a node, also known as a neuron or perceptron, has one or more weighted input connections, a transfer function that somehow mixes the inputs, and an output connection.

Node layers make up artificial neural networks (ANNs), which have an input layer, one or more hidden layers, and an output layer. Each artificial neuron, or node, has a threshold and weight associated with it and is connected to other artificial neurons. Any node whose output rises above the specified threshold value is activated and starts sending information to the top layer of the network. Otherwise, no data is sent to the next tier of the network. After the neural network has been created using training data, the weights are modified using the "Back Propagation of Errors" or BPE method. In order to offer the least amount of loss, the weights in this approach are adjusted in a certain way after each iteration. Each artificial neuron, or node, is connected to others and has a weight and threshold that go with it. Any node whose output exceeds the defined threshold value is activated and begins providing data to the network's uppermost layer. Otherwise, no data is transmitted to the network's next tier. For our purposes, the weights are changed using the "Back Propagation of Errors" approach after the neural network has been built using training data. The weights in this strategy are changed to minimize loss after each repetition.

However, to use a neural network for curve fitting, the right set of hyperparameters (activation function, number of hidden layers, number of hidden layer nodes, learning rate) must be determined. These parameters aid in the creation of a model that would provide the best fit and minimal loss. Sometimes, the selection of a certain hyperparameter has a significant impact on the loss and can be crucial when creating a neural network model for our unique situation. Trial and error can be time-consuming and uses more computer power. This issue might be greatly simplified by an analytical function that could recommend the required number of hidden layers and the number of nodes within each one.

In this work, we studied the effect of these hyperparameters on the curve fitting problem and come up with several significant conclusions which would get us close to designing an analytical function that would be a better fit to the respective curves.

Table 1. Full form of in-text abbreviations

Abbreviation	**Full Form**
PES	Potential Energy Surface
LJ	Lennard- Jonnes
MP	Morse Potentials
NNs	Neural Networks
ANNs	Artificial Neural Networks
BPE	Back Propagation Error
ELU	Exponential Linear Unit

SELU	Scaled Exponential Linear Unit
ReLU	Rectified Linear Unit
Tanh	Hyperbolic Tangent
AF	Activations Functions
NN	Neural Networks

II. LITERATURE REVIEW

Building a suitable mathematical function that has the best fit for a series of data points is the process of "curve fitting" [5], [6], [7]. The fitted curves can be used to summarize the relationships between two or more variables and to infer the values of a function in the absence of data. The curve fitting function is typically expressed as y =f(x) [3]. The polynomial of nth order, where n is the dimension of the data set, serves as the curve's representation for multidimensional data sets. Finding the proper coefficients for the multi-dimensional polynomial thus becomes the solution to the curve fitting problem [8]. Many curve fitting issues have been solved using Artificial Neural Networks (ANNs). ANNs typically exhibit higher performance capabilities than more conventional approaches [9]. ANNs are quicker and may learn from their experiences to improve their performance and adapt to their surroundings. They are made up of numerous algorithms that are frequently employed on diverse streams for issues like categorization, prediction, and machine learning [57], [58].

Backpropagation, a well-known supervised learning technique, is commonly used to train artificial neural networks to make precise predictions [10], [11]. In order to reduce the error between the target value and the actual output value, it continuously updates the weight values that are established by input-output mappings. The weight values are iteratively calculated using the gradient descent technique [12]. Utilizing activation functions, the result of the (n-1)th layer is passed on to the nth layer. There are many different activation processes, which can be classified as linear or nonlinear. Linear functions are denoted in mathematics by the formula f(x) = axe + b [13]. Sigmoid, Tanh, or Leaky, Rectified Linear Unit, and Hyperbolic Tangent ReLU are examples of nonlinear activation processes [14]. Non-linearity greatly aids in the creation of the graph smooth and facilitates generalization by the model or modifications using a range of data, and distinguishing between the results [15].

Neural Network

The basic building blocks of the neurological system are neurons, which are nerve cells. Dendrites and axon terminals in the brain serve as connections between neurons. In the majority of neurons, a voltage gradient that results in an all-or-nothing pulse known as an action potential initiates the signal traveling from the dendrites and soma to the axons. Neural networks (NN) are built on this neuronal mechanism. An entity known as the node, which is comparable to animal neurons, is specified in NN. A number of nodes are linked together and exchange information with one another (See Figure 3). The NN is made up of a network of nodes [59]. There is a weight and a direction to every link. The way the nodes are connected allows the NN to tackle a variety of issues, including curve fitting. The input layer, hidden layer(s), and an output layer of NN typically comprise three sets of nodes, and information flow takes place in this specific order [16].

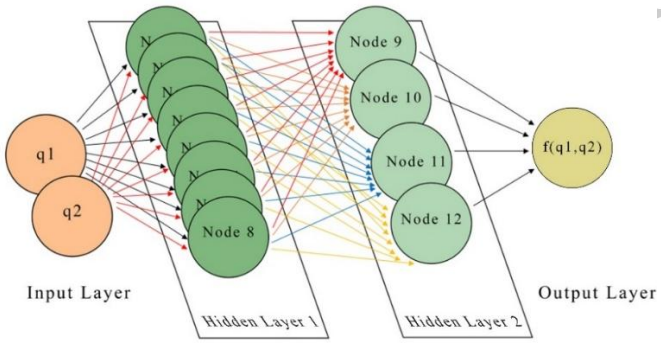

Figure 3. Schematic diagram of a NN developed for a 2-Dimensional Function

ML Approaches in Computational Materials Science

ML is employed in many disciplines, including science, engineering, and social science, and it could be useful in many future stages of the development process. Machine learning is the process of computer learning without human input. For instance, in geostatistics, ML has been used to calculate the concentration or quality of ore in relation to mine location [17]. Without human oversight, handwritten records can be converted into a digital format using pattern recognition [18]. In order to find new medications and forecast their behavior, biologists employ machine learning (ML). Other applications of ML include forecasting the weather [19], classifying data from remote sensors [20], and predicting traffic [21]. There are several uses for ML in chemistry. It has been applied to create force fields [22], predict and correlate material properties [23], and find new targeted compounds [24]. Particularly in response dynamics, ML has been utilized to analyze data from dynamic simulations and estimate the potential energy surface (PES) [25-27].

Several sorts of ML system types can be created using various standards. Depending on whether it can train progressively, it can be classified as either an online or a batch system [28–30]. A machine learning system is classed as an instance-based model if it predicts outcomes by contrasting fresh data with archived data. On the other hand, a system is referred to as model-based if

a predictive model is built using training data [31]. Whether ML systems require human supervision is another method to classify them. Based on these criteria, ML systems can be divided into four categories: reinforced, semi-supervised, unsupervised, and supervised [32–34].

When labels are only provided for a portion of the data, semi-supervised models could also be employed in computational materials science studies. In the case of property prediction problems, for instance, results for unlabeled data can be predicted with a fair amount of accuracy once a model has been trained on the labeled data. Robotics, self-driving automobiles, and board games have been where reinforcement learning has been most widely applied. Further research must be done on how this model might be used to computational materials science. Training instance, hypothesis, hyperparameter, cost function, feature, and target are a few of the notations used in ML and ANN. The input training data are represented by the training instance. The training dataset consists of a collection of training cases. The hypothesis is the solution to a particular mathematical problem. The model's performance is measured by the cost function, which can be optimized by changing the hyperparameters. The input values for ML models come from features, which are frequently referred to as descriptors or input neurons in ANN. The model's learned prediction of one or more values is the target, which corresponds to the output neurons in an ANN. The complexity of the chosen hypothesis should be comparable to the complexity of the underlying data for the best generalization. Underfitting will be a problem if the hypothesis is insufficient to explain the data [35–37].On the other hand, if the selected hypothesis is extremely complex, the model will overfit since it will learn from both the data's inherent trend and noise [38–40]. When selecting a model, it is important to carefully take into account the complexity of the hypothesis, the complexity of the training data, and the generalization performance on new examples [41–43].

Machine Learning PESs

For creating PESs or determining other attributes of unidentified chemicals or structures, machine learning (ML) techniques have grown in popularity recently [44–47]. Such methods enable computers to discover patterns in data without explicit programming [48], i.e., it is not necessary to add chemical knowledge to an ML model. For instance, there is no need to make assumptions about bonding patterns. For PES construction, acceptable reference data, which are often derived from ab initio calculations, include energy, forces, or both. Nonparametric and not constrained to a particular functional form, ML-based PESs are. The majority of machine learning techniques used to build PES are either kernel-based or rely on artificial neural networks (NNs). Both variations make use of the ability to linearize many nonlinear issues, such as forecasting energy from nuclear locations, by mapping the input to a (typically higher-dimensional) feature space [49]. The kernel trick [50–52], which enables operating in an implicit feature space without explicitly computing the coordinates of data in that space, is used by kernel-based methods to operate in that space.

III. METHODOLOGY

We have chosen the one-dimensional Morse and Lennard-Jonnes potential to solve our curve fitting problem. After achieving favourable results, we switched to the 2-dimensional PES and tried to determine the best match for the 2-D Morse potential (See Figures 2(a) and 2(b)). We have made an effort to fit the PES using a neural network model. Various hidden layer numbers and neuronal topologies were used in the tests. The hidden layers of the neural network function abstractly. A set of weighted inputs are applied to each hidden layer neuron, and an activation function is used to produce an output. We have chosen the one-dimensional Morse and Lennard-Jonnes potential to solve our curve fitting problem. After achieving favorable results, we switched to the 2-dimensional PES and tried to determine the best match for the 2-D Morse potential (See Figures 2(a) and 2(b)). We have made an effort to fit the PES using a neural network model. Various hidden layer numbers and neuronal topologies were used in the tests. The hidden layers of the neural network function abstractly. A set of weighted inputs are applied to each hidden layer neuron, and an activation function is used to produce an output. The activation functions that are employed have the following definitions:

We also looked into how our outcomes changed with various learning rates and activation functions in order to account for all the controllable variables (AF). The learning rate hyperparameter regulates how much to change the model each time the weights are updated in response to the expected error. It frequently fluctuates between 0.0 and 1 with very little positive value. By computing the weighted total and applying a bias to it, activation functions are essential in deciding whether a neuron should be engaged or not. They also provide non-linearity to a neuron's output.

Here, α is 1.0

$$\mathrm{ELU} = \begin{cases} x, & x > 0 \\ \alpha(\exp(x) - 1), & x \le 0 \end{cases}$$

$$\mathrm{SELU} = \begin{cases} \lambda x, & x > 0 \\ \lambda\alpha(\exp(x) - 1), & x \le 0 \end{cases}$$

Here, α (1.67326324) and λ (1.05070098) are pre-defined constants.

$$\mathrm{ReLU} = \max(x, 0)$$

$$\mathrm{Tanh} = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

The input is transferred to the hidden layer, the next layer in a neural network, during training. As processing takes place in the hidden layers through a system of weighted connections, nodes in the hidden layer combine data from the input layer with a set of coefficients and apply suitable weights to inputs. The activation function of each node is then multiplied by the total of these input-weight products to determine how far a signal must travel through the network before it affects the output. The

hidden layers are linked to the output layer, which is where the outputs are retrieved. This projected output is compared to the actual output in order to determine the prediction error. The magnitude of error indicates how wrong we are in our prediction.

The training method aims to constantly update these weights in order to lower the loss value. To do this, the "Back Propagation of Errors Method" is employed. Backpropagation occurs when an error spreads from the output back to the inputs; hence, the name. The neural network receives one data point at a time while having access to the entire training set. This is what we refer to as one era. We break the epoch into multiple smaller batches since it is too big to be sent to the computer all at once and updating the weights with a single pass or one epoch is insufficient. As the number of epochs increases, the neural network's weight is changed more frequently, and the curve moves from an underfitting to an optimal to an overfitting state. Once the training step is finished, the neural network may be used to predict the value of our function at an upcoming unknown point.

Our model is trained with 10,000 training epochs and assessed for under- and overfitting. The remaining 80% of the data are used for training, while the remaining 20% are used for validation. In order to assess the model's precision, we applied a loss function. Since N is the total number of training points, the loss is equal to the mean square error in this instance.

$$\text{Loss} = \frac{\sum_{i=1}^{N} (y\text{true} - y\text{pred})^2}{N}$$

All these hyperparameters are varied and the model is trained with each combination. A loop is designed to execute the desired logic. The different number of neurons in the hidden layer, learning rates, and activation functions that are tried are mentioned below:

- Hidden layer 1: [5, 10, 20, 30, 40, 50]
- Hidden layer 2: [5, 10, 20, 30, 40, 50]
- Hidden layer 3: [5, 10, 20, 30, 40, 50]
- Learning rates: [0.1, 0.01, 0.001, 0.0001]
- Activation Function: [SELU, ELU, ReLU, Tanh]

IV. RESULTS

The model is trained using all the different combinations as discussed in the above methodology section. This section will be representing the significant inferences and results for the combination which showed the best fit and minimum loss throughout the model.

Morse Potential

For the fitting of the Morse potential the neural network is trained with a minimum of one hidden layer and a maximum of three hidden layers. The number of neurons in each hidden layer is varied according to the earlier mentioned loop hyperparameters and each combination is checked for best fit and minimum value of a loss (See Figure 4 and Figure 5). The best results displayed below are obtained using the ReLU activation function and a combination of two hidden layers (See Table 2).

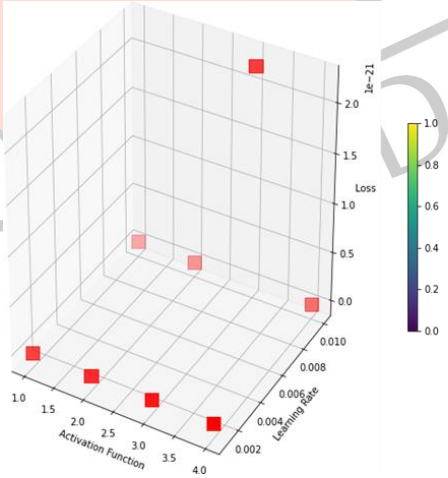

Figure 4. Variation of loss with change in activation function and learning rate. 1: SELU, 2: ELU, 3: ReLU, 4: Tanh

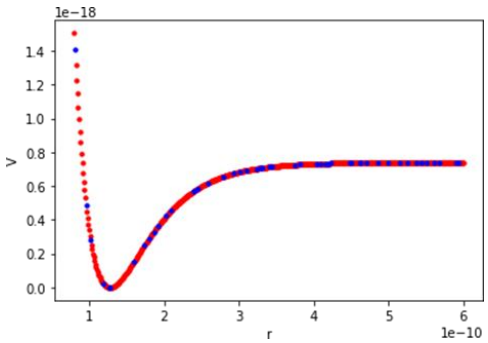

Figure 5. Fitted Morse potential using ReLU activation function with 5 nodes in each hidden layer with a learning rate of 0.01

Table 2. Matrix representing the loss with different combination of nodes in hidden layer. Columns represent hidden layer 1 and rows represent hidden layer 2. Activation function - ReLU, Learning rate - 0.01

Nodes	5	10	20	30	40	50
5	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37
10	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37
20	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37
30	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37
40	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37
50	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37	4.13e-37

Leonard Jonnes Potential

For the fitting of the Lennard Jonnes potential the neural network is trained with a minimum of one hidden layer and a maximum of three hidden layers. The number of neurons in each hidden layer is varied according to the earlier mentioned loop hyperparameters and each combination is checked for best (See Figure 6).

A fit and minimum value of a loss. For this particular potential, there is no specific combination that provides the best results (See Table 3) and we have displayed the combinations that gave better fit and minimum lossreating PESs or determining other attributes of unidentified chemicals or structures, machine learning (ML) techniques have grown in popularity recently [44–47]. Such methods enable computers to discover patterns in data without explicit programming [48].

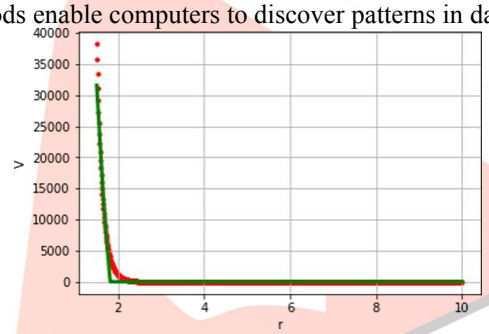

Figure 6. Fitted Lennard Jonnes potential using ReLU activation function with 100 nodes in hidden layer 1 and 50 nodes in hidden layer 2 with a learning rate of 0.1.

Table 3. Best results using different combinations, where the learning rate in all the trials is 0.1

Activation Function	Nodes in First Hidden Layer	Nodes in Second Hidden Layer	Loss
ReLU	5	5	0.0116
ReLU	50	25	1.25e-4
ReLU	50	50	5.63e-5
ReLU	100	50	2.35e-5
ReLU	500	250	1.04e-4
ReLU	500	500	7.77e-5
ELU	100	50	1.57e-4
SELU	100	50	8.71e-4

2D Morse Potential

For the fitting of the 2D Morse potential the neural network is trained with a minimum of one hidden layer and a maximum of three hidden layers. The number of neurons in each hidden layer is varied according to the earlier mentioned loop hyper-parameters and each combination is checked for best fit and minimum value of a loss (See Figure 7 and Figure 8). The best results (See Table 3) are displayed below are obtained using the ELU activation function and a combination of two hidden layers with 10 nodes and 50 nodes respectively.

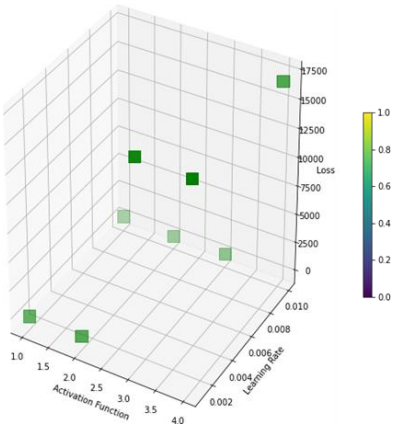

Figure 7. Variation of loss with change in activation function and learning rate. 1: SELU, 2: ELU, 3: ReLU, 4: Tanh

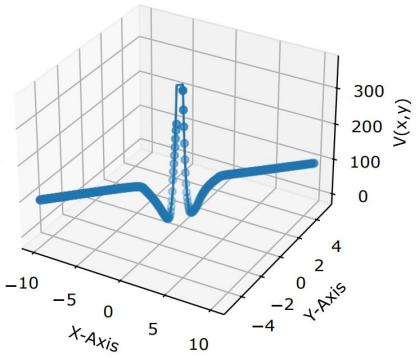

Figure 8. Fitted 2D Morse potential using ELU activation function with 40 nodes in hidden layer 1 and 10 nodes in hidden layer 2 with a learning rate of 0.01

Table 4. Matrix representing the loss with different combinations of nodes in the hidden layer. Columns represent hidden layer 1 and rows represent hidden layer 2. Activation function - ELU, Learning rate of 0.01

Nodes	**5**	**10**	**20**	**30**	**40**	**50**
5	4096.60	3.62	8085.57	3987.36	7.61	0.85
10	3.86	5.01	96.55	14.15	0.11	4043.13
20	0.52	9.17	19.79	6.19	0.92	5.52
30	11.76	0.48	0.50	0.49	1.04	1.74
40	8.81	9.76	3989.19	2.20	17.02	3.76
50	5.47	57.13	12.13	0.74	1.41	180.81

Double 2D Morse Potential

For the fitting of the double 2D Morse potential the neural network is trained with a minimum of one hidden layer and maximum of three hidden layers. The number of neurons in each hidden layer is varied according to the earlier mentioned loop hyperparameters and each combination is checked for best fit and minimum value of a loss (See Figure 9 and Figure 10). The best results displayed below are obtained using ELU activation function and a combination of two hidden layers with 10 nodes and 50 nodes respectively (See Table 5).

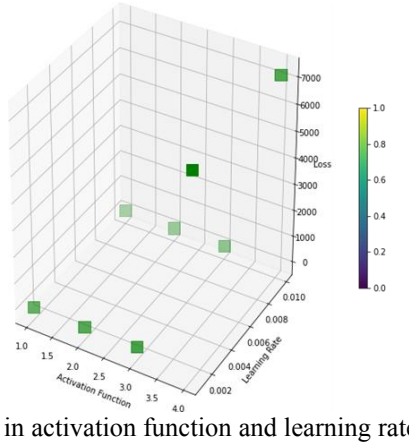

Figure 9. Variation of loss with change in activation function and learning rate. 1: SELU, 2: ELU, 3: ReLU, 4: Tanh

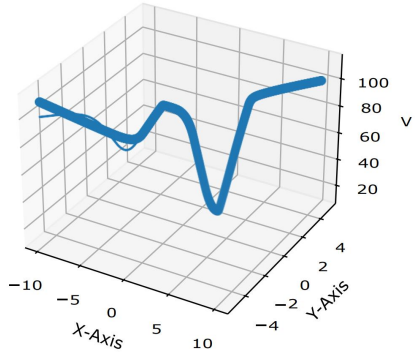

Figure 10. Fitted Double 2D Morse potential using ELU activation function with 40 nodes in hidden layer 1 and 40 nodes in hidden layer 2 with a learning rate of 0.01

Table 5. Matrix representing the loss with different combinations of nodes in the hidden layer. Columns represent hidden layer 1 and rows represent hidden layer 2. Activation function - ELU, Learning rate of 0.001

Nodes	**5**	**10**	**20**	**30**	**40**	**50**
5	1.92	5.09	9.21	5.78	3.96	4.56
10	7.96	8.51	8.67	0.51	0.57	1.10
20	5.54	8.03	0.85	0.07	0.04	0.06
30	0.37	0.01	0.02	0.08	0.01	1.81
40	0.14	0.04	0.07	0.12	0.007	0.07
50	0.03	8.13	0.05	0.11	0.12	0.03

V. CONCLUSION

The potential energy surface (PES) shows how the position of an atom affects the potential energy of a chemical system. Curve fitting has become a crucial application of machine learning because all it requires is finding a function G(q) that will help us figure out the value of our function F(q) at an unknown q. We improve the way we characterise complex potentials and expand them into multiple dimensions by fitting curves with a neural network. In order to help with the development of an analytical function, we attempt to examine how these hyperparameters affect the curve fitting situation in our study. To resolve our curve fitting issue, we have picked the 1-dimensional Morse and Lennard-Jonnes potential. Exponential linear unit (ELU), scaled exponential linear unit (SELU), rectified linear unit (ReLU), and hyperbolic tangent were used as various forms of activation functions to perform separate trails utilising varied hidden layer counts and combinations of neurons (Tanh). Our model was trained using an 80:20 train-to-test ratio over 10,000 iterations.

- We can infer the following crucial findings regarding this study as a result:
- For both 2D Morse Potential and Double 2D Morse Potential, best results using ELU activation are obtained with using 2 hidden layers of 10 and 50 nodes respectively.
- The ELU activation function worked best for 2-dimensional curves and RELU was more suitable for 1-dimensional curves.
- 0.0001 learning rates are the learning rates at which models are trying to fit better.
- Optimal results are obtained with two hidden layers in all the cases, i.e., in Lennard Jonnes, 2D Morse and Double 2D Morse potential.
- The 1-dimensional curves show little or no variation in the loss with change in the number of nodes in the hidden layer. However, the change in activation function and learning rate has a significant impact on the loss.
- The 2-dimensional curves show a change in loss with the change in the number of nodes in hidden layers.

Multidimensional PESs are a useful tool for carrying out high-quality atomistic simulations in gas- and condensed-phase environments. Another noteworthy choice that has emerged is the use of NN-based PESs in this investigation. Despite these developments, it is challenging to apply these techniques to situations that span numerous dimensions. It would be interesting to investigate simulations that portray fluctuations in local chemistry or atomic charges rather than having to exactly express them as a geometric function. Simulations that can capture changes in local chemistry or atomic charges are future possibilities that do not require explicit parametrization as a function of geometry [55], [56]. Additionally, by harnessing the computing power of current quantum computers to produce significantly huge volumes of ab initio data, current machine learning methods, atomic mechanics, and force field modelling, could exceed the constraints of previous methodologies.

VI. ACKNOWLEDGMENT

We would like to thank Professor U. Lourderaj for his supervision, and supporter S. Verma during this project, as well as NISER, Bhubaneswar, for allowing us to collaborate on this topic.

REFERENCES

[1] N.Sathyamurthy, Computational fitting of ab initio potential energy surface, 0167-7977/85/Computer Physics Reports 3 (1985) l-70.

[2] Carpenter, B. K., Ezra, G. S., Farantos, S. C., Kramer, Z. C., & Wiggins, S. (2017). Empirical classification of trajectory data: An opportunity for the use of machine learning in molecular dynamics. The Journal of Physical Chemistry B, 122(13), 3230-3241.

[3] Y.P.Varshni, Comparative Study of Potential Energy Functions for Diatomic Molecules, Reviews of Modern Physics Volume 29, 4 October 1957.

[4] Manzhos, S., & Carrington Jr, T. (2020). Neural network potential energy surfaces for small molecules and reactions. Chemical Reviews, 121(16), 10187-10217.

[5] Arlinghaus, S. (1994). Practical handbook of curve fitting. CRC press.

[6] Kolb, W. M. (1984). Curve fitting for programmable calculators. Imtec.

[7] Al Bataineh, A., & Kaur, D. (2018, July). A comparative study of different curve fitting algorithms in artificial neural network using housing dataset. In Naecon 2018-ieee national aerospace and electronics conference (pp. 174-178). IEEE.

[8] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

[9] Bishop, C. M., & Roach, C. M. (1992). Fast curve fitting using neural networks. Review of scientific instruments, 63(10), 4450-4456.

[10] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[11] Nielsen, M. A. (2015). Neural networks and deep learning (Vol. 25). San Francisco, CA, USA: Determination press.

[12] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. nature, 323(6088), 533-536.

[13] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning (pp. 448-456). PMLR.

[14] Agostinelli, F., Hoffman, M., Sadowski, P., & Baldi, P. (2014). Learning activation functions to improve deep neural networks. arXiv preprint arXiv:1412.6830.

[15] Agarap, A. F. M. (2018, February). A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. In Proceedings of the 2018 10th international conference on machine learning and computing (pp. 26-30).

[16] Biswas, R., Rashmi, R., & Lourderaj, U. (2020). Machine Learning in Chemical Dynamics. Resonance, 25(1), 59-75.

[17] Krige, D. G., & Magri, E. J. (1982). Geostatistical case studies of the advantages of lognormal-de Wijsian kriging with mean for a base metal mine and a gold mine. Journal of the International Association for Mathematical Geology, 14(6), 547-555.

[18] Indurkhya, N., & Damerau, F. J. (2010). Handbook of natural language processing. Chapman and Hall/CRC.

[19] Rasouli, K., Hsieh, W. W., & Cannon, A. J. (2012). Daily streamflow forecasting by machine learning methods with weather and climate inputs. Journal of Hydrology, 414, 284-293.

[20] Maxwell, A. E., Warner, T. A., & Fang, F. (2018). Implementation of machine-learning classification in remote sensing: An applied review. International Journal of Remote Sensing, 39(9), 2784-2817.

[21] Li, L., He, S., Zhang, J., & Ran, B. (2016). Short - term highway traffic flow prediction based on a hybrid strategy considering temporal − spatial information. Journal of Advanced Transportation, 50(8), 2029-2040.

[22] R Jinnouchi and R Asahi, Predicting catalytic activity of nanoparticles by a dftaided machine-learning algorithm, J. Phys. Chem. Lett., Vol.8, pp.4279–4283, 2017

[23] Jinnouchi, R., & Asahi, R. (2017). Predicting catalytic activity of nanoparticles by a DFT-aided machine-learning algorithm. The journal of physical chemistry letters, 8(17), 4279-4283.

[24] Yao, K., Herr, J. E., Brown, S. N., & Parkhill, J. (2017). Intrinsic bond energies from a bonds-in-molecules neural network. The journal of physical chemistry letters, 8(12), 2689-2694.

[25] Botu, V., Batra, R., Chapman, J., & Ramprasad, R. (2017). Machine learning force fields: construction, validation, and outlook. The Journal of Physical Chemistry C, 121(1), 511-522.

[26] Cui, J., & Krems, R. V. (2016). Efficient non-parametric fitting of potential energy surfaces for polyatomic molecules with Gaussian processes. Journal of Physics B: Atomic, Molecular and Optical Physics, 49(22), 224001.

[27] Blank, T. B., Brown, S. D., Calhoun, A. W., & Doren, D. J. (1995). Neural network models of potential energy surfaces. The Journal of chemical physics, 103(10), 4129-4137.

[28] Xu, Q., & Dai, X. (2008, July). Online learning ANN-Inversion excitation controller of the multi-machine power system. In 2008 Chinese Control and Decision Conference (pp. 758-763). IEEE.

[29] Mu, S., & Tian, S. (2006, November). A novel online learning algorithm of support vector machines. In 2006 8th international Conference on Signal Processing (Vol. 3). IEEE.

[30] Ananthakrishnan S, Prasad R, Stallard D, Natarajan P. Batch-mode semi-supervised active learning for statistical machine translation. Comput Speech Lang. 2013;27:397–406.

[31] Zubair, O. A., Ji, W., & Weilert, T. E. (2017). Modeling the impact of urban landscape change on urban wetlands using similarity weighted instance-based machine learning and Markov model. Sustainability, 9(12), 2223.

[32] Nielsen RD. Introduction to machine learning for digital library applications. Jcdl'18: Proceedings of the 18th ACM/IEEE Joint Conference on Digital Libraries; 2018:421-422.

[33] Saravanan R, Sujatha P. A state of art techniques on machine learning algorithms: A perspective of supervised learning approaches in data classification. Proceedings of the 2018 Second International Conference on Intelligent Computing and Control Systems (Iciccs); 2018: 945-949.
[34] Zhang, Y., & Yang, Q. (2018). An overview of multi-task learning. National Science Review, 5(1), 30-43.
[35] Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proceedings of the National Academy of Sciences, 116(32), 15849-15854.
[36] Li, W., Liang, P., & Hu, J. (2019). An autoencoder - based piecewise linear model for nonlinear classification using quasilinear support vector machines. IEEJ Transactions on Electrical and Electronic Engineering, 14(8), 1236-1243.
[37] Fujii, K., Suzuki, C., & Hasuo, M. (2019). Robust regression for automatic fusion plasma analysis based on generative modeling. IEEE Transactions on Plasma Science, 47(7), 3305-3314.
[38] Gossmann, A., Pezeshk, A., & Sahiner, B. (2018, March). Test data reuse for evaluation of adaptive machine learning algorithms: over-fitting to a fixed'test'dataset and a potential solution. In Medical Imaging 2018: Image Perception, Observer Performance, and Technology Assessment (Vol. 10577, pp. 121-132). SPIE.
[39] Nakatsu, R. T. (2017, July). Information visualizations used to avoid the problem of overfitting in supervised machine learning. In International Conference on HCI in Business, Government, and Organizations (pp. 373-385). Springer, Cham.
[40] Dietterich, T. (1995). Overfitting and undercomputing in machine learning. ACM computing surveys (CSUR), 27(3), 326-327.
[41] Wang, D., Wang, P., & Ji, Y. (2015). An oscillation bound of the generalization performance of extreme learning machine and corresponding analysis. Neurocomputing, 151, 883-890.
[42] Takeda, A., & Kanamori, T. (2014). Using financial risk measures for analyzing generalization performance of machine learning models. Neural networks, 57, 29-38.
[43] Zou, B., Li, L., & Xu, J. (2006, July). The generalization performance of learning machine with NA dependent sequence. In International Conference on Rough Sets and Knowledge Technology (pp. 568-573). Springer, Berlin, Heidelberg.
[44] Rupp, M., Tkatchenko, A., Müller, K. R., & Von Lilienfeld, O. A. (2012). Fast and accurate modeling of molecular atomization energies with machine learning. Physical review letters, 108(5), 058301.
[45] Montavon, G., Rupp, M., Gobre, V., Vazquez-Mayagoitia, A., Hansen, K., Tkatchenko, A., ... & Von Lilienfeld, O. A. (2013). Machine learning of molecular electronic properties in chemical compound space. New Journal of Physics, 15(9), 095003.
[46] Hansen, K., Montavon, G., Biegler, F., Fazli, S., Rupp, M., Scheffler, M., ... & Mü ller, K. R. (2013). Assessment and validation of machine learning methods for predicting molecular atomization energies. Journal of Chemical Theory and Computation, 9(8), 3404-3419.
[47] Hansen, K., Biegler, F., Ramakrishnan, R., Pronobis, W., Von Lilienfeld, O. A., Mü ller, K. R., & Tkatchenko, A. (2015). Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space. The journal of physical chemistry letters, 6(12), 2326-2331.
[48] Samuel, A. L. (2000). Some studies in machine learning using the game of checkers. IBM Journal of research and development, 44(1.2), 206-226.
[49] Schölkopf, B., Smola, A., & Müller, K. R. (1997, October). Kernel principal component analysis. In International conference on artificial neural networks (pp. 583-588). Springer, Berlin, Heidelberg.
[50] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152).
[51] Schölkopf, B., Smola, A., & Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. Neural computation, 10(5), 1299-1319.
[52] Unke, O. T., Koner, D., Patra, S., Käser, S., & Meuwly, M. (2020). High-dimensional potential energy surfaces for molecular simulations: from empiricism to machine learning. Machine Learning: Science and Technology, 1(1), 013001.
[53] Stahl N, Falkman G, Karlsson A, Mathiason G, Bostrom J. Deep convolutional neural networks for the prediction of molecular properties: Challenges and opportunities connected to the data. J Integr Bioinform. 2019;16:20180065.
[54] Goh, GB, Siegel, C, Vishnu, A, Hodas, NO, Baker, N. Chemception: A deep neural network with minimal chemistry knowledge matches the performance of expert-developed QSAR/QSPR models. arXiv.org; 2017.
[55] Unke, O. T., Koner, D., Patra, S., Käser, S., & Meuwly, M. (2020). High-dimensional potential energy surfaces for molecular simulations: from empiricism to machine learning. Machine Learning: Science and Technology, 1(1), 013001.
[56] Handley, C. M., & Popelier, P. L. (2010). Potential energy surfaces fitted by artificial neural networks. The Journal of Physical Chemistry A, 114(10), 3371-3383.
[57] Imam, R., Anwer, F., & Nadeem, M. (2022). An Effective and enhanced RSA based Public Key Encryption Scheme (XRSA). International Journal of Information Technology, 14(5), 2645-2656.
[58] Anas, M., Imam, R., & Anwer, F. (2022, January). Elliptic Curve Cryptography in Cloud Security: A Survey. In 2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 112-117). IEEE.
[59] Areeb, Q. M., Imam, R., Fatima, N., & Nadeem, M. (2021, October). AI Art Critic: Artistic Classification of Poster Images using Neural Networks. In 2021 International Conference on Data Analytics for Business and Industry (ICDABI) (pp. 37-41). IEEE.


IJEDR