CHIN. PHYS. LETT. Vol. 38, No. 5 (2021) 050701

Machine Learning Kinetic Energy Functional for a One-Dimensional Periodic System

Hong-Bin Ren(任宏斌)[1,2], Lei Wang(王磊)[1,3], and Xi Dai(戴希)[4*]

[1]*Beijing National Laboratory for Condensed Matter Physics and Institute of Physics, Chinese Academy of Sciences, Beijing 100190, China*

[2]*School of Physical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China*

[3]*Songshan Lake Materials Laboratory, Dongguan 523808, China*

[4]*Department of Physics, Hong Kong University of Science and Technology, Kowloon 999077, Hong Kong, China*

(Received 26 January 2021; accepted 9 March 2021; published online 2 May 2021)

Kinetic energy (KE) functional is crucial to speed up density functional theory calculation. However, deriving it accurately through traditional physics reasoning is challenging. We develop a generally applicable KE functional estimator for a one-dimensional (1D) extended system using a machine learning method. Our end-to-end solution combines the dimensionality reduction method with the Gaussian process regression, and simple scaling method to adapt to various 1D lattices. In addition to reaching chemical accuracy in KE calculation, our estimator also performs well on KE functional derivative prediction. Integrating this machine learning KE functional into the current orbital free density functional theory scheme is able to provide us with expected ground state electron density.

DOI: 10.1088/0256-307X/38/5/050701

Density functional theory (DFT)[1] has been widely applied in chemistry, biotechnology and material sciences to theoretically explore desired properties of new compounds.[2,3] Despite its great success in solid state physics and quantum chemistry, current DFT implementation is not computationally efficient. In particular, it has $O(M^3)$ complexity (M is proportional to the number of orbitals used in calculation), which makes the simulation process time consuming and limits its usage in a large system with more than a thousand of atoms. During the recent decade, great success has been reached by applying machine learning (ML) technology to understand and translate natural language,[4] generating and decoding the complex audio signals,[5] inferring features from real world image and video.[6] To date, ML technology has also been applied to the field of new material search and some encouraging results have already been achieved.[7−16]

Naturally, ML algorithms are an interpolation method. In practice, most of the ML algorithms cannot fully utilize the symmetries of the particular systems and they fail to generalize well beyond their training data, which indicates that special care is needed to be made for different types of materials. Instead of trying to learn physics rules directly, we want to combine ML with components from the traditional simulation method, where the laws of physics are automatically satisfied and allow our ML algorithm to focus on what they do best, learning optimal rules for interpolating in complex, high-dimensional spaces.

In Ref. [17], Snyder *et al.* have demonstrated the idea of using ML to find kinetic energy functional for the orbital free density functional theory (OFDFT) method. The OFDFT[18,19] can be regarded as a direct implementation of the Hohenberg Kohn theorem. Instead of solving Schrödinger's equation, it iteratively solves the Euler–Lagrange (EL) equation, which makes it manageable in time when dealing with large systems. Although by now we have some candidates for exchange correlation functional, no counterpart exists for KE functional, which strongly limits the accuracy and generalization of the OFDFT. In Ref. [17], Snyder *et al.* used kernel ridge regression (KRR) to fit the non-interacting KE functional using a dataset that is calculated from a simple quantum chemistry problem where several non-interacting fermions are confined in a one-dimensional (1D) box. In predicting the KE, they are able to outperform several widely used human designed functionals with their ML functional on this particular problem.

Inspired by their success, we decide to focus on the KE functional for solid state physics, where the system is infinite and periodic. From an ML perspective, because the KE functional maps the electron density function into a scalar, and the KE functional derivative maps the electron density function into another function, their targets are not the same. Therefore, we think of predicting KE and its functional deriva-

Supported by the Hong Kong Research Grants Council (Project No. GRF16300918), the National Key R&D Program of China (Grant Nos. 2016YFA0300603 and 2016YFA0302400), and the National Natural Science Foundation of China (Grant No. 11774398).

*Corresponding author. Email: daix@ust.hk

© 2021 Chinese Physical Society and IOP Publishing Ltd

tive as two tasks. As is shown in Fig. 1, our ML KE functional contains two parts, one for predicting KE, and the other for predicting the KE functional derivative. In the first part, we use the Gaussian process (GP) method to map electron density function to its KE. We choose the GP because it is as powerful as KRR used by Snyder *et al.* , and it allows us to determine the value of hyperparameters via the gradient based optimization method. In the second part, we use another GP whose covariance matrix is the Hessian matrix of covariance matrix of the first GP to map electron density function to KE functional derivative. We can avoid significant error resulted from directly differentiating ML KE functional by explicitly learning the functional derivative.

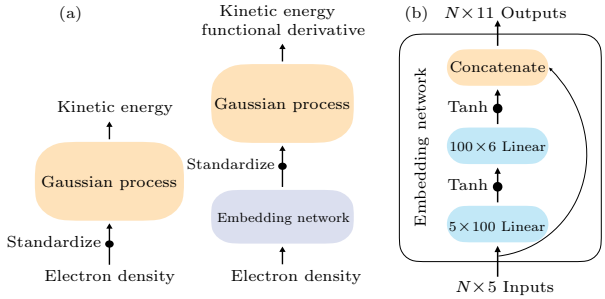

Fig. 1. (a) Left hand: KE estimator. Electron densities are standardized and passed to the GP to evaluate KE. Right hand: KE functional derivative estimator. Electron densities are first embedded in a $\mathbb{R}^{11}$ manifold. The embedded features are then standardized and passed to the GP to evaluate KE functional derivative. (b) Structure of embedding network. N is the number of data samples within our dataset. We use black dot to represent operation with no learnable parameters, and use colored rectangular to represents operation with learnable parameters.

Our dataset contains 1000 electron density, KE, and KE functional derivative data solved from Schrödinger's equation on 1D simple lattice with various periodic external potential $V(x)$.

$$-\frac{\hbar^2}{2m_e}\frac{d^2}{dx^2}\psi(x)+V(x)\psi(x)=E\psi(x), \tag{1}$$

where $\psi(x)$ is the wavefunction, which has an eigenvalue E; m_e is the mass of electron; and $\hbar$ is the reduced Plank constant.

Noticing that the 1D lattice is fully characterized by its lattice constant a_0, we write Eq. (1) in a unit system where $a_0=1$, the energy unit $E_0=\hbar^2/(m_e a_0^2)$. Solving the rescaled equation then provides us with electron densities [Fig. 2(a)] and KEs that are independent of specific lattice structure. Though this rescaling trick is simple, it ensures that our ML model for KE functional can be generalized to various 1D lattices.

External potential $V(x)$ is generated from its Fourier transform in momentum space:

$$V(\mathcal{G})=\begin{cases}0, & \mathcal{G}/2\pi>10,\\ -\sqrt{2\pi}\sum_{i=1}^{n}b_i\sigma_i\exp\{-\frac{1}{2}(\sigma_i\mathcal{G})^2\}\cos(\mu_i\mathcal{G}), & \mathcal{G}/2\pi\le 10,\end{cases} \tag{2}$$

where $\mathcal{G}$ is the reciprocal lattice vector that satisfies $\mathcal{G}/2\pi\in\mathbb{Z}$; b_i, σ_i and μ_i are parameters that control the shape of $V(x)$. We choose $n=5$ and uniformly sample $b_i\in[1,10]$, $\sigma_i\in[0.01,0.3]$, and $\mu_i\in[0.3,0.7]$1000 times. The resulting $V(x)$'s are shown in Fig. 2(b).

The KE functional derivative data is obtained via the EL equation:

$$\delta_\rho T\equiv\frac{\delta T}{\delta\rho(x)}=\mu-V(x), \tag{3}$$

where T is the KE functional, and μ is the chemical potential which is used to produce required electron number per cell. Since each electron density $\rho(x)$ in our dataset is the ground state electron density of a particular $V(x)$. Equation (3) holds for every electron density in our dataset, and the KE derivative is thus determined by the negative external potential up to a constant.

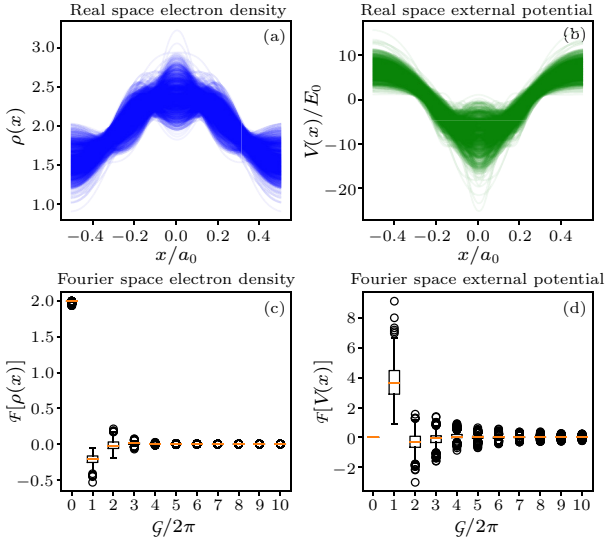

Fig. 2. (a) Electron density functions in our DFT dataset. The electron number per unit cell is set to 2. (b) Behavior of 1000 external potential functions used to generate our DFT dataset. Since $V(x)$ is determined up to a constant. We have choose a gauge such that integral of $V(x)$ over $[0,1)$ is zero. (c) Boxplot of Fourier components of electron density function. (d) Boxplot of Fourier components of external potential function.

Unlike Ref. [17] in which the authors directly fed discretized $\rho(x)$ to their ML model, we start from the plane wave expansion method in solid state physics. We first apply Fourier transform (FT) to $\rho(x)$, and then use the obtained Fourier components $\mathcal{F}[\rho(x)](\mathcal{G})$ as features. From an ML point of view, FT can be viewed as a dimensionality reduction technique. As is

shown in Fig. 2(c), each $\rho(x)$ in Fig. 2(a) can be accurately described by its first 5 Fourier components. This suggests that FT can effectively transform data from several hundreds of dimensional Euclidean spaces to $\mathbb{R}^5$ without losing much information. In statistical ML, learning on high-dimensional data is problematic due to the curse of dimensionality.[20] FT addresses this problem and helps to reduce the impact of high-dimensional data on our ML model.

We denote the training data by $\boldsymbol{X}$ and $\boldsymbol{Y}$ for features and targets, and use $\mathcal{X}$ and $\mathcal{Y}$ to represent the feature space and target space. The GP model can be learned by solving the following linear equation:

$$(\boldsymbol{K}_\theta + \sigma^2 \boldsymbol{I})\boldsymbol{\alpha} = \boldsymbol{Y}, \tag{4}$$

where $\boldsymbol{K}_\theta$ is the covariance matrix of multivariate Gaussian distribution, $\boldsymbol{I}$ is the identity matrix, and σ^2 is the regularization factor used to control the magnitude of $\boldsymbol{\alpha}$. The nonlinear function $F : \mathcal{X} \to \mathcal{Y}$ can be constructed from the representer theorem:[21]

$$F(\boldsymbol{x}_*) = \sum_{i=1}^{N} \boldsymbol{\alpha}_i k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_i), \tag{5}$$

where $\boldsymbol{\alpha}_i \in \boldsymbol{\alpha}$; $\boldsymbol{x}_i$ stands for the ith data sample in $\boldsymbol{X}$; $\boldsymbol{x}_* \in \mathcal{X}$ is a vector outside $\boldsymbol{X}$; N is the number of training examples in $\boldsymbol{X}$; and $k_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv (\boldsymbol{K}_\theta)_{ij}$. Here θ is the hyperparameter of the covariance matrix, and can be learned by minimizing the negative log-likelihood (NLL) of GP:

$$\mathrm{NLL}(\theta) = \frac{1}{2}\boldsymbol{Y}^T(\boldsymbol{K}_\theta + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{Y} + \frac{1}{2}\log\det(\boldsymbol{K}_\theta + \sigma^2\boldsymbol{I}). \tag{6}$$

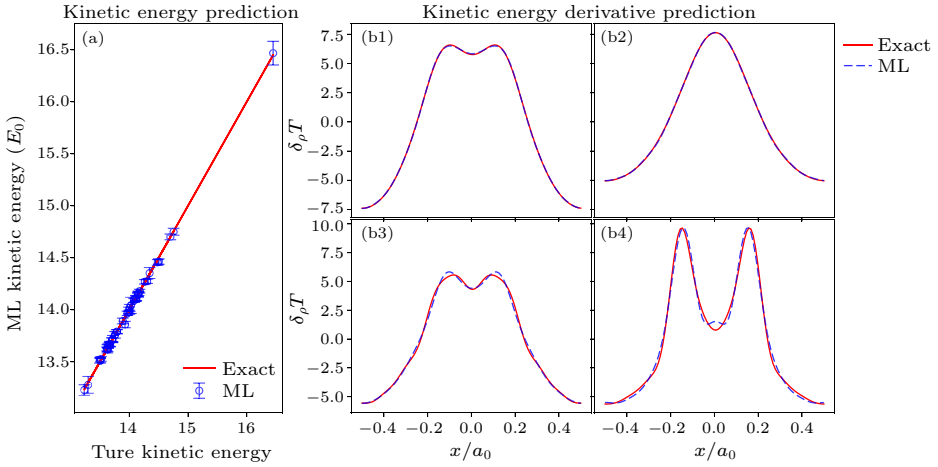

Fig. 3. (a) KE computed from exact electron density in testing dataset by our ML model. Error bar represents the uncertainty of the ML model about the predicted value. (b1)–(b4) Several KE functional derivative computed from exact electron density in testing dataset by our ML model. The exact values of $\delta_\rho T$ are represented by the red line, our prediction is represented by the dashed blue line.

We randomly select 800 out of 1000 data to build the training dataset, select another 100 data to be validation dataset, and use the remaining 100 data as the testing dataset. To train the first part of our ML model, we let $\boldsymbol{X}$, $\boldsymbol{Y}$ to be electron densities and KEs; therefore, $\mathcal{X} \subseteq \mathbb{R}^5$, $\mathcal{Y} \subseteq \mathbb{R}$. We set $\boldsymbol{K}$ to be the anisotropic Matérn type covariance matrix:

$$\boldsymbol{k}_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j) = \lambda^2(1 + \sqrt{5}\boldsymbol{d} + \frac{5}{3}\boldsymbol{d}^2)\exp(-\sqrt{5}\boldsymbol{d}),$$
$$\boldsymbol{d} \equiv \sqrt{\sum_{k=1}^{5}\left(\frac{\boldsymbol{x}_{i,k} - \boldsymbol{x}_{j,k}}{l_k}\right)^2}, \tag{7}$$

where λ and l_k's belong to θ, and $\boldsymbol{x}_{i,k}$ is the kth component of $\boldsymbol{x}_i$. We drawn λ and l_k's randomly from $(0, 1]$, and update their values with the L-BFGS method during minimization of Eq. (6). The training process converges after 5 iterations. KEs for testing dataset are calculated from Eq. (5) and compared to the exact KE in Fig. 3(a). The mean square error (MSE) between our prediction and exact values is less than 0.1099 kcal/mol, which is far below the chemical accuracy (1 kcal/mol).

The second part of the ML model is different from the first part since its target is KE functional derivative. We let KE functional derivatives to be $\boldsymbol{Y}$ so that $\mathcal{Y} \subseteq \mathbb{R}^{11}$ [see Fig. 2(d)]. Then, following the work of Alvarez *et al.* .[22] We flatten $\boldsymbol{Y}$ to a vector in $\mathbb{R}^{11\,N}$, and reshape the Hessian of Eq. (7) to be the covariance matrix:

$$(\boldsymbol{K}_{\mathrm{Hess}})_{ik,jl} \equiv \frac{\partial^2 \boldsymbol{k}_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\partial \boldsymbol{x}_{i,k} \partial \boldsymbol{x}_{j,l}}, \tag{8}$$

where ik and jl are composite indices that represent the reshaped row and column indexes of $\boldsymbol{K}_{\mathrm{Hess}}$, respectively.

Since the dimensionality values of $\mathcal{X}$ and $\mathcal{Y}$ in the KE functional derivative estimation task are not the same, we cannot directly build GP on $\boldsymbol{X}$ and $\boldsymbol{Y}$. We

design an embedding network [Fig. 1(b)] to transform features in $\mathbb{R}^5$ to the $\mathbb{R}^{11}$ space. Each $\boldsymbol{x}_i \in \boldsymbol{X}$ will be passed through two linear layers (without bias) with a tanh activation function, and then be concatenated to the original 5-dimensional input to build an 11-dimensional feature.

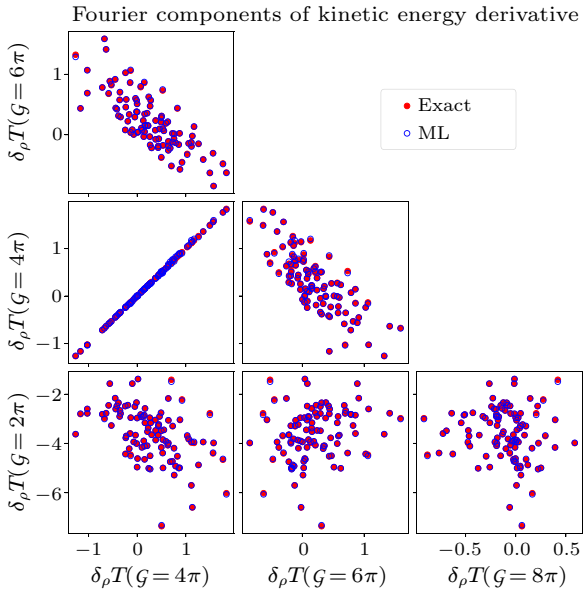

Fig. 4. Pairwise scatter plot of major Fourier components of $\delta_\rho T$ calculated by our ML model on testing dataset.

According to Mackay[23] and Calandra *et al.* ,[24] covariance matrix together with embedding network can be viewed as a new covariance matrix whose hyperparameter θ includes parameters of the embedding network. Therefore, we can use Eq. (6) to jointly train the embedding network and GP. We use the L-BFGS method to train our model. The derivative of NLL(θ) with respect to θ is obtained by Pytorch's *Autograd* module. After 50 iterations of training on a single P100 GPU, we obtain the KE functional derivative model with MSE less than 2.537 kcal/mol on testing dataset. Several examples of predicted KE functional derivative on testing dataset are shown in Figs. 3(b1)–3(b4).

As soon as we have a relatively accurate KE functional, we can find the ground state electron density for a given external potential by solving Eq. (3). Directly solving this fixed point equation requires the second-order functional derivative of the KE functional to be evaluated, which is infeasible for our ML KE functional. In practice, we consider the equivalent constrained minimization problem:

$$\begin{aligned} \min_{\rho(x)} \; & T[\rho(x)] + \int_0^1 V(x)\rho(x)\boldsymbol{d}x, \\ \text{s.t.} \quad & \forall x \in [0,1),\ \rho(x) \geq 0, \\ & \int_0^1 \rho(x)\boldsymbol{d}x = n_\mathrm{e}, \end{aligned} \tag{9}$$

where n_e is the number of electrons per unit cell. We remove the particle number constrain in Eq. (9) by introducing a normalization prefactor:

$$\rho(x) \leftarrow \frac{n_\mathrm{e}}{\int_0^1 \rho(x)\boldsymbol{d}x}\rho(x), \tag{10}$$

and change variable from $\rho(x)$ to $\phi(x) \equiv \sqrt{\rho(x)}$[18] to maintain the positiveness of electron density function during the minimization.

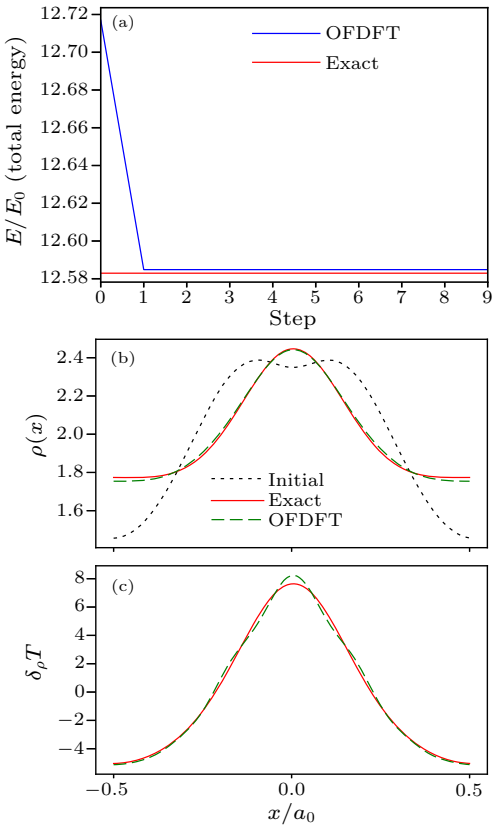

Fig. 5. (a) Blue line represents energy decay in solving the EL equation in OFDFT. Red line is the exact ground state energy of the corresponding external potential calculated by solving Schrödinger's equation. (b) Variation of electron density in solving the EL equation. We start from an initial guess (the black-dotted line) and arrive at the OFDFT ground state electron density (the green-dashed line), which overlaps the exact ground state electron density (the red line). (c) Functional derivative of kinetic energy functional calculated from the OFDFT ground state electron density (the green-dashed line) and exact function derivative (the red line).

We use the L-BFGS method to solve Eq. (9), and show the approximate ground state electron density obtained from the minimum of the ML energy functional and KE functional derivative evaluated at that electron density in Figs. 5(b) and 5(c). Since we have used FT to reduce the dimensionality of our data, we do not need to use local PCA introduced in Ref. [17] to stabilize the KE functional derivative calculation along the optimization process. We must mention that the initial value of electron density should be carefully selected in order to obtain a reasonable approximate ground state electron density. Some initial electron

densities may lead the optimization to areas that are not covered by the training dataset and this makes the prediction of our ML model unreliable. From Fig. 5(a), we find that after a single iteration, the total energy of our ML OFDFT method reaches the minimum that is approximately 1.19 kcal/mol above the true ground state energy. This energy difference is larger compared to the KE prediction error mentioned above, and is comparable to the KE functional derivative prediction error. This is because the KE functional derivative error will be accumulated to approximated electron density obtained from Eq. (9), and we are evaluating the energy from this density instead of the actual one.

In summary, we have developed an ML algorithm to learn kinetic energy functional that is applicable to all the 1D system under a periodic boundary condition. Because 1D lattices are characterized by their lattice constant a_0, we introduce a rescaling method to map them to a prototype 1D model whose unit cell length is set to 1. We build a dataset of electron density, KE and KE functional derivative by solving Schrödinger's equation for potentials generated from Eq. (2). We use FT to transform each real space electron density function to a vector in $\mathbb{R}^5$, and design an embedding network to facilitate the learning of KE functional derivative. On KE prediction, our GP model achieves chemical accuracy on the testing dataset. Our manifold GP[24] model also performs well in KE functional derivative prediction. Finally, we replace the KE functional in OFDFT with our ML KE functional. Solving the ML based OFDFT problem then provides us with reasonable ground state electron density.

Though we are dealing with non-interacting electrons in this study, our methodology can be generalized to treat interacting electrons through the Kohn–Sham (KS) formulation of DFT. In this case, our external potential is replaced by the KS potential which includes the exchange-correlation functional. Electron density, KE and KE functional derivative in our dataset can all be calculated by prevalent DFT software such as the Vienna *ab initio* simulation package (VASP). Generalizing our method to 2D or 3D is also possible. We can use 2D or 3D Fourier transform to build the feature and then apply the same algorithm. However, the dimensionality of 2D and 3D feature is even higher, suggesting that more powerful methods such as neural network can be used instead of GP for KE functional derivative estimation.

Rescaling Method. Given Schrödinger's equation (1), we let $z = x/a_0$, and the equation becomes:

$$-\frac{1}{2}\frac{d^2}{dz^2}\tilde{\psi}(z) + \tilde{V}(z)\tilde{\psi}(z) = \frac{E}{(\hbar^2/m_e a_0^2)}\tilde{\psi}(z), \quad (11)$$

where $\tilde{\psi}(z) \equiv \psi(x)$. Set energy unit $E_0 = \hbar^2/m_e a_0^2$, and let $\tilde{V}(z) \equiv V(x)/E_0$, $\varepsilon \equiv E/E_0$, we finally arrive at an equation:

$$-\frac{1}{2}\frac{d^2}{dz^2}\tilde{\psi}(z) + \tilde{V}(z)\tilde{\psi}(z) = \varepsilon\tilde{\psi}(z), \quad (12)$$

whose solution is independent of a_0.

Model Selection—Training Dataset and Training Steps. The accuracy of an ML model strongly depends on the training dataset and training process.

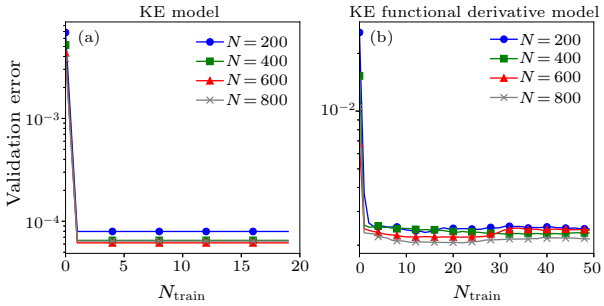

Fig. 6. Validation error with respect to size of training dataset and number of training iterations: (a) training of the KE model, (b) training of the KE functional derivative model.

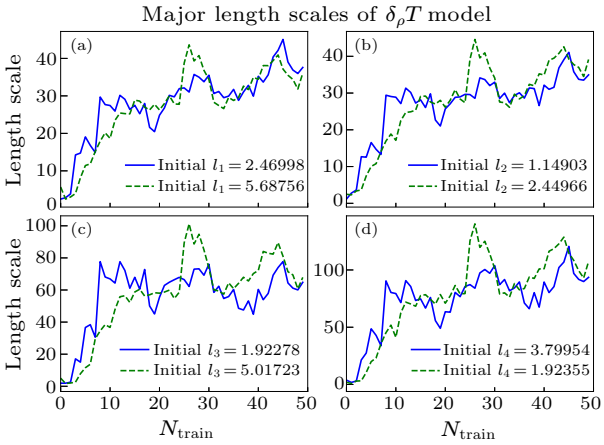

Fig. 7. Variation of length scales of covariance matrix of the KE functional derivative model. We initialize each length scale in two different values, and train the model on 800 training data for 50 iterations.

Table 1. Initial and final values of length scale of covariance matrix of the KE functional model. The initial values are randomly sampled, and the final values are obtained after 20 iterations of training on 800 training data.

	Initial value	Final value
l_0	0.95311481	4.31956064
	10.95311432	4.31957246
l_1	0.56069778	3.73920018
	10.56069773	3.73922276
l_2	0.3535394	4.51812732
	10.35353942	4.51815901

In Fig. 6, we show the validation error obtained by training our ML model on different size datasets. We find a clear data driven pattern, and we also note that our models converge pretty quick. The valida-

tion error of the KE model will no longer decrease after 5 steps of training. For the KE functional derivative model, approximately 50 steps are needed for the training process to converge.

Covariance Function and Its Parameter. The performance of GP largely depends on the covariance matrix used. In this study, we follow the recommendation of Stein[25] and use the Matérn type covariance function in Eq. (7) for the first part of our KE functional model. From Table 1, we find that several major length scales of covariance matrix will take on values that are relatively independent of the their initial values after training. This does not strictly hold for the second part of our model, whereas in Fig. 7 we show that the evolution of length scales follow the same trend.

References

[1] Kohn W and Sham L J 1965 *Phys. Rev.* **140** A1133
[2] Hafner J, Wolverton C and Ceder G 2006 *MRS Bull.* **31** 659
[3] Jones R O 2015 *Rev. Mod. Phys.* **87** 897
[4] Wu Y, Schuster M, Chen Z, Le Q V and Norouzi M 2016 arXiv:1609.08144 [cs.CL]
[5] Hinton G, Deng L, Yu D, Dahl G and Mohamed A R 2012 *IEEE Signal Process. Mag.* **29** 29
[6] Krizhevsky A, Sutskever I and Hinton G E 2012 *Imagenet Classification with Deep Convolutional Neural Networks* in *Advances in Neural Information Processing Systems* ed Pereira F, Burges C J C, Bottou L and Weinberger K Q (Curran Associates, Inc.) vol 25 pp 1097–1105
[7] Behler J and Parrinello M 2007 *Phys. Rev. Lett.* **98** 146401
[8] Bartók A P, Payne M C, Kondor R and Csányi G 2010 *Phys. Rev. Lett.* **104** 136403
[9] Rupp M, Tkatchenko A, Müller K R and Von Lilienfeld O A 2012 *Phys. Rev. Lett.* **108** 058301
[10] Zhang L, Han J, Wang H, Car R and Weinan E 2018 *Phys. Rev. Lett.* **120** 143001
[11] Ramakrishnan R, Dral P O, Rupp M and von Lilienfeld O A 2014 *Sci. Data* **1** 140022
[12] Gilmer J, Schoenholz S S, Riley P F, Vinyals O and Dahl G E 2017 *Proceedings of the 34th International Conference on Machine Learning* (ICML'17) vol 70 pp 1263–1272
[13] Chen G, Chen P, Hsieh C Y, Lee C K and Liao B 2019 arXiv:1906.09427 [cs.LG]
[14] Zhuang L, Ye Q, Pan D and Li X Z 2020 *Chin. Phys. Lett.* **37** 043101
[15] Yao T S, Tang C Y, Yang M, Zhu K J and Yan D Y 2019 *Chin. Phys. Lett.* **36** 068101
[16] Tang Q, Yang J H, Liu Z P and Gong X G 2020 *Chin. Phys. Lett.* **37** 096802
[17] Snyder J C, Rupp M, Hansen K, Müller K R and Burke K 2012 *Phys. Rev. Lett.* **108** 253002
[18] Lignères V L and Carter E A 2005 *Handbook of Materials Modeling* (Berlin: Springer) p 137
[19] Witt W C, Beatriz G, Dieterich J M and Carter E A 2018 *J. Mater. Res.* **33** 777
[20] Bengio Y, Delalleau O and Le R N 2005 *Département d'Informatique et Recherche Opérationnelle, Université de Montréal, Canada* Tech. Rep. 1258
[21] Schölkopf B, Herbrich R and Smola A J 2001 *International Conference on Computational Learning Theory (COLT 2001)* in *Lecture Notes in Computer Science* (Berlin: Springer) vol 2111 pp 416–426
[22] Alvarez M A, Rosasco L and Lawrence N D 2011 arXiv:1106.6251 [stat.ML]
[23] MacKay D J 1998 *NATO ASI Ser. F: Comput. Syst. Sci.* **168** 133
[24] Calandra R, Peters J, Rasmussen C E and Deisenroth M P 2016 *2016 International Joint Conference on Neural Networks (IJCNN)* (24-29 July 2016, Vancouver, BC, Canada) pp 3338–3345
[25] Stein M L 2012 *Interpolation of Spatial Data: Some Theory for Kriging* (Berlin: Springer)