Nonempirical shape dynamics of heavy nuclei with multitask deep learning

N. Hizawa and K. Hagino
Department of Physics, Kyoto University, Kyoto 606-8502, Japan

(Received 17 October 2023; accepted 7 December 2023; published 9 January 2024)

A microscopic description of nuclear fission represents one of the most challenging problems in nuclear theory. While phenomenological coordinates, such as multipole moments, have often been employed to describe fission, it is not obvious whether these parameters fully reflect the shape dynamics of interest. We here propose a novel method to extract collective coordinates, which are free from phenomenology, based on multitask deep learning in conjunction with density functional theory (DFT). To this end, we first introduce randomly generated external fields to a Skyrme energy density functional (EDF) and construct a set of nuclear number densities and binding energies for deformed states of ^{236}U around the ground state. By training a neural network on such a dataset with a combination of an autoencoder and supervised learning, we successfully identify a two-dimensional latent variables that accurately reproduce both the energies and the densities of the original Skyrme-EDF calculations, within a mean absolute error of 113 keV for the energies. In contrast, when multipole moments are used as latent variables for training in constructing the decoders, we find that the training data for the binding energies are reproduced only within 2 MeV. This implies that conventional multipole moments do not provide fully adequate variables for a shape dynamics of heavy nuclei.

DOI: 10.1103/PhysRevC.109.014312

I. INTRODUCTION

It is a highly nontrivial issue to determine the most consistent collective coordinates to describe a shape dynamics of heavy nuclei, such as nuclear fission. A related question is how many collective coordinates one would need to achieve an essential description of such shape dynamics. These are the questions which we discuss in this paper.

Nuclear fission is in fact one of the most fundamental and yet challenging problems in nuclear theory. In most calculations carried out so far, except for the ones based on the time-dependent density functional theory (TDDFT) [1], one usually assumes a few collective coordinates for fission and constructs a (multidimensional) potential energy surface [2]. For spontaneous fission, one in addition computes the moment of inertia and estimates the decay half-life based on the Wentzel-Kramers- Brillouin (WKB) approximation [3–8]. On the other hand, for induced fission, one computes transport coefficients and solves the dynamics with, e.g., the Langevin approach [9,10], the random walk approach [11,12], and the time-dependent generator coordinate method (TDGCM) [13–15].

The most serious problem in these approaches is that one has to assume *a priori* relevant collective degrees of freedom for fission based on a phenomenological consideration, despite the collective degrees of freedom for large amplitude motions, such as fission, being highly nontrivial. Therefore, it is not obvious whether chosen collective coordinates are optimum for the dynamics of interest. In principle, the self-consistent collective coordinate (SCC) method [16–18] can be used to derive a collective path without resorting to a phenomenological assumption. Even though several approximate calculations based on SCC have been carried out [19,20], it is still computationally quite demanding. Evidently, a more efficient method to determine collective coordinates is urged.

We believe that a machine learning technique will provide a useful means in this respect. As it continues to evolve, machine learning is bringing about a pivotal shift in nuclear theory. Currently, regression analysis with supervised learning is a common practice in nuclear theory [21–34]. For example, it has been employed to construct a new mass table [35]. It is important to notice that machine learning extends beyond supervised learning, encompassing other techniques such as unsupervised learning [36], reinforcement learning [37], and self-supervised learning [38]. These have successfully been applied not only in physics but also in several other disciplines as well.

Of particular interest within unsupervised learning is the method of using an autoencoder [39]. In this method, input data are mapped onto low-dimensional latent variables, with which the input data are subsequently reconstructed back. The underlying idea of this method is based on the manifold hypothesis [40], that is, data of interest occupy only limited locations in a quite high-dimensional space. This technique has been applied in various contexts, including a data characteristic analysis and a computation cost reduction.

In nuclear physics, Ref. [41] recently employed autoencoders to acquire low-dimensional representations of quadrupole and octupole deformed Hartree-Fock-Bogoliubov (HFB) wave functions, which were to be used in generator coordinate method (GCM) calculations. Notice that the manifold hypothesis may hold rather trivially in the study of Ref. [41], as autoencoders were applied only to limited datasets. Nevertheless, the basic idea of Ref. [41] remains intriguing. See also Ref. [42] for a recent related work. In fact, the concept of collective coordinates in nuclear physics is practically equivalent to the manifold hypothesis. That is, both of them state that low-energy dynamics of a nucleus can be approximated within a restricted region in the infinite-dimensional manifold which fully specifies degrees of freedom of a system, such as the full Hilbert space. It would thus be reasonable to anticipate that machine learning methods, including unsupervised learning, work effectively in describing nuclear collective motions.

In this paper, we shall apply multitask learning (MTL) [25,33,43] to discuss adiabatic shape dynamics of ^{236}U near the ground state. To this end, in this work we employ the density functional theory (DFT) [44,45] and combine an autoencoder and supervised learning. A remarkable advantage of DFT is that the only dynamical variable is in principle a particle number density. This is in contrast to other many-body theories with many-body wave functions, which are significantly more difficult to handle even numerically. By employing multitask learning, we shall extract a common feature representation of the energy and the density of ^{236}U, that is, common latent variables which yield both the energy and the density. This representation is expected to contain a large amount of information on the adiabatic dynamics.

The paper is organized as follows. In Sec. II, we will detail our procedure for MTL. To construct a dataset, we will introduce random potentials to the system. In Sec. III, we will discuss criteria for evaluations of our MTL calculations. We will apply these criteria to discuss the applicability of the conventional collective coordinates for fission, by comparing the results with the criteria. In Sec. IV, we will carry out MTL and discuss latent variables for shape dynamics. We will then summarize the paper in Sec. V and discuss future perspectives.

II. FORMULATION

A. Skyrme EDF with random external potentials

Our purpose in this paper is to analyze a low-energy deformation dynamics of ^{236}U with the density functional theory (DFT) [44,45] and multitask learning. The DFT evaluates a ground state dynamics, i.e., the adiabatic dynamics, under arbitrary external fields. An elemental degree of freedom in the DFT is the particle number density ρ, with which any observable, including the ground state energy E, can be evaluated. Namely, in principle, there is an energy density functional (EDF) $E[\rho]$, that solely depends on the particle number density, to describe the adiabatic dynamics. However, in nuclear physics, almost all calculations employ a Kohn-Sham type energy density functional (KS-EDF), which depends not only on the particle number density but also on other densities, such as the kinetic energy, the spin-orbit, and the pairing densities [46]. Among them, the Skyrme EDF is one of the most famous and commonly used KS-EDFs [46–48], and we shall also adopt it in this paper. For simplicity, we impose the axial and the time reversal symmetries on the system. The actual calculations are performed with the computer code HFBTHO (v3.00) [49]. This is an open source DFT solver, in which Kohn-Sham wave functions are expanded on an axial symmetric harmonic oscillator basis,

$$\varphi_{n_z,n_r,\Lambda}(\boldsymbol{r};b_z,b_\perp)=\varphi_{n_z}(\xi;b_z)\varphi^{|\Lambda|}_{n_r;b_\perp}(\eta)\frac{e^{i\Lambda\phi}}{\sqrt{2\pi}}, \tag{1}$$

$$\varphi_{n_z}(\xi;b_z)=\frac{1}{\sqrt{b_z2^{n_z}n_z!\sqrt{\pi}}}H_{n_z}(\xi)e^{-\xi^2/2}, \tag{2}$$

$$\varphi^{|\Lambda|}_{n_r}(\eta;b_\perp)=\sqrt{\frac{2n_r!}{b_\perp^2(n_r+|\Lambda|)!}}\,\eta^{|\Lambda|/2}L^{|\Lambda|}_{n_r}(\eta)e^{-\eta/2}, \tag{3}$$

$$\xi=\frac{z}{b_z},\quad \eta=\left(\frac{r_\rho}{b_\perp}\right)^2, \tag{4}$$

where $\boldsymbol{r}=(r_\rho,z,\phi)$ is a three-dimensional (3D) cylindrical coordinate, and $b_\perp$ and b_z are the oscillator length parameters. H_{n_z} and $L^{|\Lambda|}_{n_r}$ are the Hermite polynomials and the associated Laguerre polynomials, respectively. Note that the octupole deformation is included in our analysis, as we do not impose the reflection symmetry along the z axis.

We adopt the SLy4 parameter set [50] and a surface type paring interaction [46] to evaluate the Kohn-Sham DFT within the Hartree-Fock-Bogoliubov (HFB) scheme [48]. The strengths of the pairing interaction are determined so that the average pairing gaps of protons and neutrons agree with the empirical value $12/\sqrt{A}$ MeV with the mass number $A=236$. The resultant values are -650.98 MeV/fm^3 for protons and -526.53 MeV/fm^3 for neutrons. We also optimize b_z and $b_\perp$ to minimize the ground state energy of ^{236}U, with the maximum number of oscillator shells N_{tot} of 26, which is large enough to describe the deformation dynamics around the ground state of ^{236}U. We obtain $b_z=2.08$ fm and $b_\perp=1.92$ fm, with which the ground state energy is found to be -1780.87 MeV.

To construct a set of the densities and the energies, (ρ,E), one must solve the Kohn-Sham equation with various external fields v. Even though the density constraint method [51] directly provides the energy for a desired density, a numerical cost in this method may increase when a numerical accuracy is required. We shall therefore adopt a random potential approach [52] in this paper.

Notice that it is computationally impossible to deal with fully arbitrary densities. Even if that was possible, such densities might contain an enormous amount of information that is not of our interest, especially that which corresponds to high energy configurations. Therefore, it is desirable to sample densities according to a probability distribution $p[\rho]$ such that a large portion of the sampled densities are relevant to our purpose. However, what one actually does in the KS-EDF is to sample external fields from some probability distribution $q[v]$ that is normalized as

$$\int\mathcal{D}v\,q[v]:=1, \tag{5}$$

with the functional integration of v.

In principle, one can map the probability distribution $q[v]$ onto the probability distribution $p[\rho]$ in the following way. Since there is a bijection $\rho[v]$ and $v[\rho]$ in the DFT, the conditional probability is formally denoted by $P(\rho|v)=\delta[\rho-\rho[v]]$, where the delta function is normalized as

$$\int\mathcal{D}\rho\,\delta[\rho-\rho[v]]:=1,\ \int\mathcal{D}\rho\,p[\rho]:=1. \tag{6}$$

Therefore, sampling v from $q[v]$ yields the probability distribution

$$p[\rho] = \int \mathcal{D}v\, \delta[\rho - \rho[v]]q[v] = \left[\left| \mathrm{Det}\left(\frac{\delta v}{\delta \rho} \right) \right| q[v] \right]_{v=v[\rho]}, \tag{7}$$

where Det is a functional determinant.

Since the relation between the densities and the external fields, that is, $\rho[v]$ and $v[\rho]$, is nontrivial, how one can choose external potentials appropriately is a highly complicated problem. Although the strategy of selecting an appropriate $q[v]$ is unknown, it is meaningful to sample v highly randomly. It is important to notice here that in general the result of machine learning with a dataset sampled from $q[v]$ may not work well with another dataset generated from a different probability distribution $q'[v]$. This is recognized as a serious issue, referred to as the problem of the domain shift, in the field of machine learning [53,54]. In particular, in the field of DFT, it has been shown that there can be a case where results trained with a *simple* $q[v]$ have little predictive ability when it is applied to highly random data with $q'[v]$ [55,56]. On the other hand, results with random data with $q'[v]$ are expected to have a much better predictive ability, and thus we employ random potentials in this study. To this end, we introduce random potentials based on the axial harmonic oscillator basis given by Eq. (1):

$$v^{(i)}(r,z) = \sum{}' \, v^{(i)}_{n_z,n_r,|\Lambda|} \varphi_{n_z}(\xi; b^{(i)}_z) \varphi^{|\Lambda|}_{n_r}(\eta; b^{(i)}_\perp), \tag{8}$$

where $\sum'$ indicates that the sum is restricted with a condition $n_z/q + 2n_r + |\Lambda| \leqslant N_{\rm shell}$, with $q = (b_z/b_\perp)^2 = 1.1736$. Here, i denotes a index of data, for which $v^{(i)}_{n_z,n_r,|\Lambda|}$, $b^{(i)}_z$, and $b^{(i)}_\perp$ are randomly generated. Note that, in addition to the weight factors $v_{n_z,n_r,|\Lambda|}$, we also randomly sample the length parameters, b_z and $b_\perp$, even though the basis functions can express arbitrary function with fixed values of b_z and $b_\perp$. This is to avoid the neural network learning the specific scales $(b_z, b_\perp)$ when adopting a finite shell number $N_{\rm shell}$. We apply the same external fields to both protons and neutrons.

The random potentials (8) are not symmetric along the z axis, and the center-of-mass position of the system can move around. To fix it, we add the quadratic constraint on the center-of-mass position to the KS-EDF during the HFB iterations:[1]

$$E_{\rm CoM}[\rho] = \frac{C}{2}\left(\int d^3r\, z\rho(\boldsymbol{r}) - z_0 \right)^2, \tag{9}$$

with $C = 1.0$ MeV fm^{-2} and $z_0 = 0$. When one needs a dataset in which the center-of-mass position is not fixed, one can simply add data by shifting the densities. Because of the translational invariance in the nuclear Hamiltonian, the total binding energy will remain the same. This approach reduces the cost of creating a new dataset and maintaining it.

[1]Although not done in this paper, an alternative way to fix the center-of-mass position is to introduce the Lagrange multiplier. This operation corresponds to restricting the domain of the random potentials.

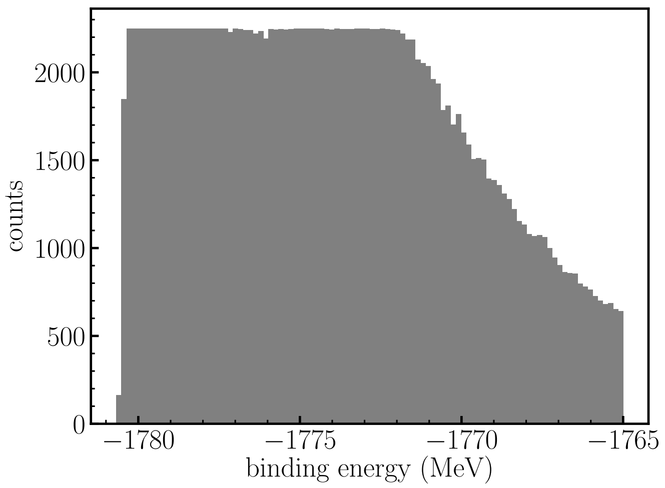

FIG. 1. A histogram of the binding energies for the dataset generated in this work. Since we randomly remove excessive amount of data, the distributional bias is largely eliminated. The ground state energy is −1780.87 MeV, and one can see that the data are almost uniformly distributed below the excitation energy of 8 MeV, i.e., the energy of −1772 MeV.

The basic strategy to determine the random potentials is to generate a large number of deformed states near the ground state, with excitation energies below about 15 MeV, which is larger than the first barrier for fission. To this end, we scale the optimized oscillator lengths as $b^{(i)}_{z,\perp} = (1 + t^{(i)}_{z,\perp})b_{z,\perp}$ with uniform random numbers of $t^{(i)}_{z,\perp}$ within [0, 0.5]. On the other hand, we individually generate $v^{(i)}_{n_z,n_r,|\Lambda|}$ for each set of $(n_r, n_z, |\Lambda|)$ using uniform random numbers within $[-v_0, v_0]$. Note that a single scale cutoff parameter v_0 leads to a significant bias in the distribution of the calculated binding energies. To avoid this problem, we adopt a multiple values of v_0, that is, 0.6, 1.2, 1.8, and 2.4 MeV fm$^{3/2}$. Moreover, a certain set of data should be removed so that the resultant data do not have a large bias in the binding energies. For this purpose, we randomly remove excessive amount of data at each bin in the histogram when the number of data exceeds 2250. The total number of bins is taken to be 100. Furthermore, we also discard datasets outside of $-1780.87 \leqslant E \leqslant -1765.00$ MeV. In addition, as we are interested in the shape dynamics only around the ground state in this paper, we reject the data with large deformation, with a cutoff of [3.0 b, 16.0 b] for an expectation value of the mass quadruple moment operator

$$\hat{Q}_{\lambda k} = r^\lambda Y_{\lambda k}(\hat{\theta}, \hat{\phi}) \tag{10}$$

with $\lambda = 2$ and $k = 0$, where $Y_{\lambda k}$ is the spherical harmonics with the polar coordinates $\boldsymbol{r} = (r, \theta, \phi)$.

With this strategy, we generate 298 982 random potentials, out of which 181 134 potentials remain for the machine learning. The composition of each v_0 is 20% for $v_0 = 0.6$ MeV fm$^{3/2}$, 23% for $v_0 = 1.2$ MeV fm$^{3/2}$, 49% for $v_0 = 1.8$ MeV fm$^{3/2}$, and 8% for $v_0 = 2.4$ MeV fm$^{3/2}$. Figures 1 and 2 show the histogram of the binding energies and the energy surface for the remaining data, respectively. For the latter, we plot the energy for each density in the

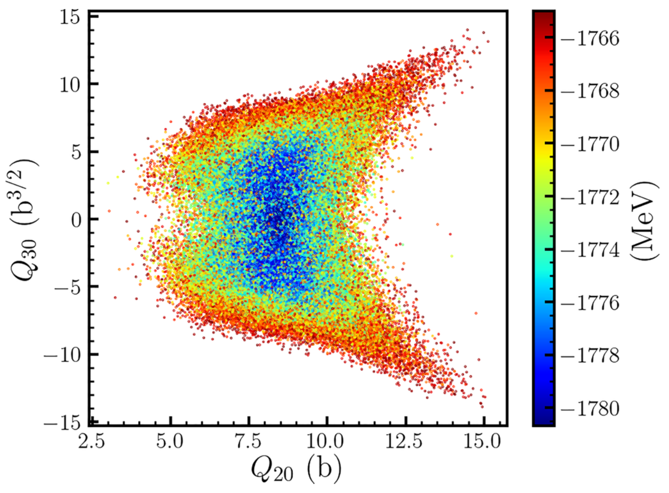

FIG. 2. A scatter plot for the quadruple and the octupole moments for the dataset. The color of each points shows the corresponding binding energy. The image appears symmetric along $Q_{30} = 0$, reflecting the fact that the results of the Skyrme-EDF are invariant under the parity transformation with respect to the z axis.

two-dimensional plane for the quadrupole and the octupole moments corresponding to the density. Note that the binding energy is calculated excluding the energy for the external fields. To utilize the densities for deep learning, we discretize them with the mesh size 0.4 fm for both the z and r axes, which is approximately half of values commonly used in Skyrme DFT calculations with a 3D lattice representation. The numbers of mesh points are 48 points (for the r axis) and 128 points (for the z axis). The nucleon density is then regarded as an monotonic image of 48 × 128 pixels. Notice that it is desirable to have the number of pixels with a power of 2, since layers that double or halve the number of vertical and horizontal pixels are often used in the field of image recognition. Our choice of mesh points approximately satisfies this condition.

B. Deep learning

1. *Multitask learning*

Based on the manifold hypothesis and the assumption of collective coordinates in nuclear theory, we expect that our dataset constructed in the previous subsection can be well characterized by a small number of parameters. To extract such latent variables of a dataset, the use of an autoencoder is a common approach [39]. We apply this to extract latent variables for our dataset. Namely, an input density is encoded to latent variables and then a decoder attempts to reproduce the original density with the latent variables as inputs. This is schematically denoted as

$$\rho \to z = \mathcal{E}[\rho] \to \rho = \mathcal{D}(z), \tag{11}$$

where $z = \{z_1, z_2, \dots, z_{d_z}\}$ are the extracted latent variables with dimension d_z, and $\mathcal{E}$ and $\mathcal{D}$ are an encoder and a decoder, respectively. The number of extracted latent variables may become too large to understand if a high accuracy is imposed onto the encoder-decoder system. In practice one would thus

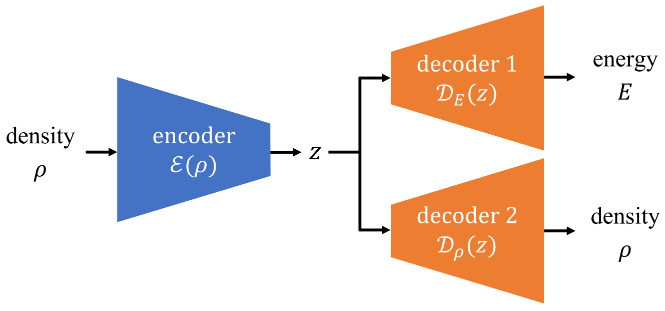

FIG. 3. A conceptual diagram of our model. The encoder $\mathcal{E}$ compresses the information on the density ρ into the latent variables z, and the decoder of the density $\mathcal{D}_\rho$ reconstructs the density itself from the latent variables. At the same time the other decoder $\mathcal{D}_E$ predicts the energy from the latent variables.

need a compromise for the accuracy. In addition, a large error may be generated when reconstructed densities are applied to predict energies with the orbital-free EDF (OF-EDF). This problem is known as the density driven error (DDE) [57], and in this case the latent variables may not have enough information on the energies.

On the other hand, one can directly apply the encoder-decoder structure to learn the OF-EDF itself. Namely, one can construct a decoder which reproduces the energy, rather than the original density, as

$$\rho \to z = \mathcal{E}[\rho] \to E = \mathcal{D}(z). \tag{12}$$

However, in this supervised learning approach, there is always a risk of leading to meaningless variables, for example, when all of the latent variables are regarded as a transformation of the energy.

To avoid such drawbacks, we combine these two tasks:

$$\rho \to z = \mathcal{E}[\rho] \to E = \mathcal{D}_E(z) \text{ and } \rho = \mathcal{D}_\rho(z), \tag{13}$$

where $\mathcal{D}_E$ and $\mathcal{D}_\rho$ are decoders for the energy and for the density, respectively. The encoder-decoder functions are approximated by neural networks. Figure 3 illustrates the conceptual architecture of our model. We can expect that the common feature representation z contains information on both the densities and the energies.

Machine learning is referred to as multitask learning (MTL) when multiple tasks are solved simultaneously [43]. While MTL is often utilized to improve the generalization performance of all tasks, the primary purpose of our study is to apply MTL to obtain a common feature representation across the tasks.

2. *Architecture*

To perform MTL, one must define a specific model for the encoder and the decoders, $\mathcal{E}$, $\mathcal{D}_E$, and $\mathcal{D}_\rho$. In this work, the input for the neural network is a nucleon density $\rho(r, z)$, that is, the binding energy is determined only by the total density, $\rho = \rho_p + \rho_n$, rather than the proton and neutron densities ρ_p and ρ_n separately. The total nucleon density $\rho(r, z)$ can be regarded as a monochromatic image with the size 1 × 48 × 128. As this size is too large to use in an architecture which consists only of fully connected layers, we mainly use convolutional

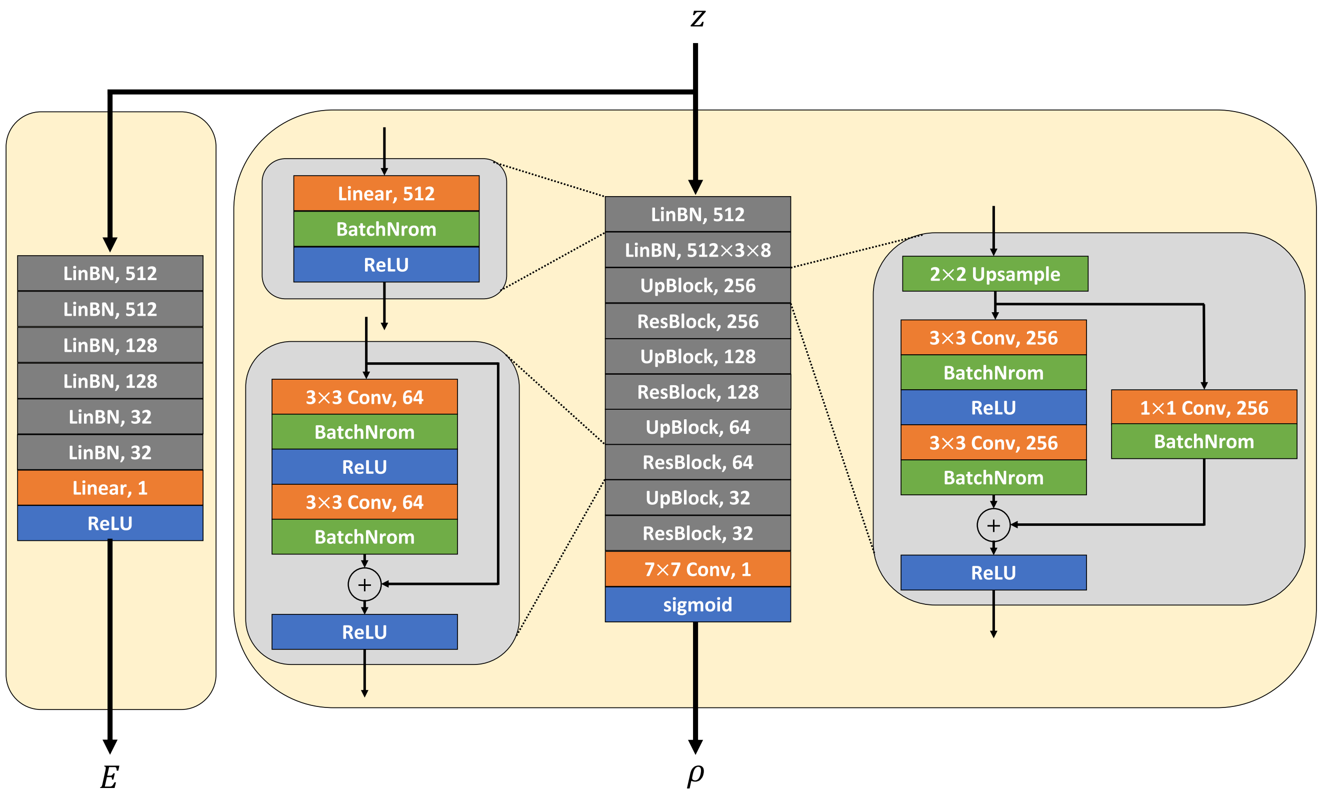

FIG. 4. The architecture of the decoders in our MTL model. The left and right blocks correspond to the decoders for the energies, $\mathcal{D}_E$, and for the densities, $\mathcal{D}_\rho$, respectively. In the convolutional layers, the left numbers denote the kernel sizes, and the right numbers are for the number of the output channels. At each convolutional layer, the amount of padding is taken such that the width and the height of the input image are equal to those of the output image. The upsample layer doubles the size of the image both vertically and horizontally, for which the interpolation is not applied.

neural networks (CNNs), which have been successful in the image recognition field. We adopt ResNet (residual neural network) in particular [58], which is one of the most famous CNN models and won a 2015 image recognition competition [59]. The critical idea of ResNet is to use the skip (residual) connection, where an output from a previous layer is added to the output of the current layer, allowing for a benefit of multilayering. In the original paper of ResNet [58], five different models with 18, 34, 50, 101, and 152 layers are proposed, which are called ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152, respectively.

For the encoder $\mathcal{E}$, we adopt the ResNet18 model. Since our image data is smaller than the one of ImageNet [60], for which the size of images is $3 \times 256 \times 256$, we remove the 2D max pooling layer with the kernel size $= 3$, stride $= 2$, and padding $= 1$ from the original ResNet18. The size of each output sample is equal to the latent dimension d_z, which is a changeable parameter.

For the decoder for the densities, $\mathcal{D}_\rho$, we do not use ResNet as it is, but adopt a model motivated by ResNet as shown in Fig. 4.[2] Note that we do not use the transposed convolution because it may cause checkerboard artifacts [62]. On the other hand, upsampling which we employ in our model does not create such problem, although one needs to pay attention to jaggies (staircase edges). In addition, the upsample layer does not explicitly make a specific direction along the z axis, which is physically reasonable for the axially symmetric system studied in this paper. For the decoder of the energy, $\mathcal{D}_E$, we use a model with six fully connected layers (see Fig. 4). Notice that we use a relatively small number of layers in this study to avoid an overcomplicated dependence of the energy on the latent variables.

[2]We also checked the parametric rectified linear init (PReLU) model [61] for the activation functions, but the performance was not improved.

3. *Loss function*

Multitask learning involves several tasks and an appropriate optimization is a critical problem. A common approach is to balance the individual loss functions for separate tasks [43]. Several approaches have been proposed to combine these different loss functions, such as uncertainty weighting (UW) [63] and dynamic weight average (DWA) [64]. In our MTL model, the energy prediction task and the density prediction task tend to compete with each other, especially when the latent dimension d_z is small. It would be desirable to define a single metric to evaluate the performances of these tasks, but it is difficult to determine it in advance. Therefore, in this paper we simply combine an energy loss $\mathcal{L}_E$ and a density loss

$\mathcal{L}_\rho$ with a constant ratio α as

$$\mathcal{L}_{\rm MTL}(D;\alpha) = \mathcal{L}_E(D) + \alpha\, \mathcal{L}_\rho(D), \tag{14}$$

with

$$\mathcal{L}_E(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}\left(E^{(i)}_{\rm pred} - E^{(i)}_{\rm true}\right)^2 \tag{15}$$

and

$$\mathcal{L}_\rho(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}\int d^3r\left(\rho^{(i)}_{\rm pred}(\boldsymbol{r}) - \rho^{(i)}_{\rm true}(\boldsymbol{r})\right)^2, \tag{16}$$

where $|D|$ is the number of data in a dataset D. The constant hyperparameter α determines the relative importance between the energy and the density for optimization. We will vary this parameter to examine the performance of our MTL, as we will show in the next section. All the parameters in the neural networks are optimized to minimize the loss function $\mathcal{L}_{\rm MTL}$. Note that in the actual calculations the data are min-max normalized, even though we return to the original physical normalization when we discuss the performance of MTL. Then the energies E and the densities ρ are also normalized, and the loss function for the energies $\mathcal{L}_E(D)$ is dimensionless, but that for the densities $\mathcal{L}_\rho(D)$ has a physical dimension fm^3 originating from the radial integral in Eq. (16). Therefore, the physical dimension of α is fm^{-3}.

4. Optimization

To optimize the parameters in the neural networks, we utilize the Adam (adaptive moment estimation) optimizer [65] with the default hyperparameters in PYTORCH [66], except for the learning rate. The initial value of the learning rate is 10^{-4} and, following Ref. [41], we dynamically reduce the learning rate by a factor of 0.5 if the loss of validation data is not improved among the last 15 epochs. One can easily implement this with “torch.optim.lr_scheduler.ReduceLROnPlateau”. When using such an approach, that is, the approach with a dynamic learning rate decay, it is not fair to evaluate the final performance with the validation data as the approach itself depends upon the data. Therefore, we randomly divide our dataset into training data (90%), validation data (5%), and test data (5%), among which the test data are utilized only for a final evaluation of the performance.

In our model, the minibatch size is 128 and the number of epochs is 1000. Since the convergence of the validation loss is worse than that of the training loss in our MTL calculations, and the validation loss sometimes fluctuates during learning, we adopt the parameters which yield the smallest loss $\mathcal{L}_{\rm MTL}$ for the validation data and use them to evaluate the results. As we will show in the next section, we will also perform another type of deep learning calculations in this paper. Even in that case, we use the same optimization method unless otherwise mentioned.

It is useful to mention the numerical cost of our calculations: optimizing our MTL model for a single set of hyperparameters takes 50 GPU hours with processor NVIDIA RTX 3090TI. Namely, we must run n NVIDIA RTX 3090TIs for $50/n$ hours.

III. CRITERIA FOR EVALUATION

A. Criterion for energy

Before we evaluate results of MTL, let us first discuss criteria for such evaluations. We first discuss errors for the energies. Because the number of our data is finite, it is natural that the energy cannot be perfectly predicted, and thus such training should not be done in terms of generalization performance. Therefore, we first estimate how much errors our dataset admits in terms of energy prediction. Namely, we learn the OF-EDF $E[\rho]$ using supervised learning, and regard the resultant error as a limitation of the energy prediction.

Table I summarizes the mean absolute errors (MAEs) and the mean square errors (MSEs) for the energy prediction with several neural networks. These are defined as

$$\mathrm{MAE} = \frac{1}{|D|}\sum_{i=1}^{|D|}\left|E^{(i)}_{\rm pred} - E^{(i)}_{\rm true}\right|, \tag{17}$$

and

$$\mathrm{MSE} = \frac{1}{|D|}\sum_{i=1}^{|D|}\left|E^{(i)}_{\rm pred} - E^{(i)}_{\rm true}\right|^2, \tag{18}$$

respectively.ResNets are the same as those described in Sec. II, except for the output layer, for which we operate ReLU (rectified linear unit) at the last stage. ResNet09, which is not included in the original paper [58], is a model in which the residual blocks are halved from ResNet18. We optimize each of the models in Table I using the MSE loss function, Eq. (15). We notice that the errors of ResNet34 and ResNet50 are almost the same as that of ResNet18, while the error of ResNet09 is significantly worse, implying that ResNet18 has an enough representational capacity to capture the characteristics of our dataset. Because of this, we mainly use ResNet18 in this paper. We thus define the criterion for the energy prediction as 64 keV.

We also evaluate the OF-EDF with the vision transformer (ViT) model [67], which has achieved state-of-the-art performance in image recognition. ViT is an application of the transformer [68], which has been remarkably successful, especially in large language models (LLMs), to image recognition. We utilize a ViT model consisting of 16 transformer blocks with multihead self-attention in which the number of heads and the dimension of hidden layers are 16 and 2048, respectively. The patch size, the embedding dimension, and the dropout rate are 8×8, 128, and 0.3, respectively. Table I indicates that the errors of our ViT model are much larger than those with the ResNets. However, it is important to point out that ViT requires a large amount of data to outperform CNN models [67]. Once a large amount of data suitable for machine learning will become available in nuclear physics, we expect that ViT will play an important role. Therefore, it is meaningful to study the performance of ViT at this stage.

The calculations presented so far are performed with the total densities as inputs. To validate such calculations, we

TABLE I. The mean absolute error (MAEs) and the mean square error (MSEs) for supervised learning of OF-EDF $E[\rho]$ with ResNets and ViT. The results imply that 64 keV is a criterion for energy predictions. For comparison, we also perform another training with proton and neutron number densities as inputs, as shown in the last row.

			Validation data (5%)		Test data (5%)	
Model	Input(s)	Output	MAE (MeV)	MSE (MeV^2)	MAE (MeV)	MSE (MeV^2)
ResNet18	ρ	E	0.0633	0.00965	0.0638	0.00963
ResNet34	ρ	E	0.0638	0.00965	0.0646	0.00949
ResNet50	ρ	E	0.0649	0.01084	0.0649	0.01023
ResNet09	ρ	E	0.0724	0.01197	0.0720	0.01161
ViT [67]	ρ	E	0.2613	0.15161	0.2704	0.16302
ResNet18	ρ_n, ρ_p	E	0.0641	0.00973	0.0635	0.00951

also perform another training with the proton and the neutron number densities as inputs. The result is shown in the last row in Table I, which indicates that the errors are similar to those with the total densities.

B. Criterion for density

Let us next evaluate the errors in the densities. It is important to notice that density is an input variable in the DFT, and supervised learning cannot be used to evaluate the errors. Therefore, we use the autoencoder (AE) [39], which is a kind of unsupervised learning. In our MTL shown in Fig. 3, if we remove the decoder for the energy, $\mathcal{D}_E$, as well as the loss function for the energy, the model is equivalent to the AE for the density [see Eq. (11)]. In this way, we evaluate the AE for several latent dimensions. The results are shown in Fig. 5. Here, the MAE for the density is defined as

$$\mathrm{MAE} = \frac{1}{|D|}\sum_{i=1}^{|D|}\int d^3r\left|\rho^{(i)}_{\mathrm{pred}}(\boldsymbol{r}) - \rho^{(i)}_{\mathrm{true}}(\boldsymbol{r})\right|. \qquad (19)$$

Note that the MAE is a dimensionless variable as the densities in this equation have physical dimension. One can see that the MAEs in lower dimensions are as large as a few percent of the mass number, $A = 236$. These large errors imply that our dataset contains a variety of densities. Because of this, increasing the latent dimension does not rapidly reduce the errors, even though the errors are significantly small when the latent dimension is large: MAE of 0.5 is achieved for $d_z = 512$, that is, the maximum dimension for the original ResNets [58]. For each d_z, we will use the corresponding MAE for a criterion for density.

C. Errors with multipole moments

Conventionally, the multipole moments $\{Q_{\lambda,k}\}$ have often been employed to describe nuclear shape dynamics. It is therefore interesting to evaluate with our dataset how much information these variables contain. To this end, we apply a supervised learning to predict the energies from several sets of multipole moments, $E(Q_{20}, Q_{30}, \dots, Q_{L0})$. A neural network which we use is the decoder of our MTL model, $\mathcal{D}_E$, with $\{Q_{\lambda,k}\}$, instead of the latent variables z, as inputs. Figure 6 shows the MAEs for several sets of the input variable defined as $S_L = \{Q_{20}, Q_{30}, \dots, Q_{L0}\}$. One can see that these errors are significantly larger than the criterion defined in Sec. III A,

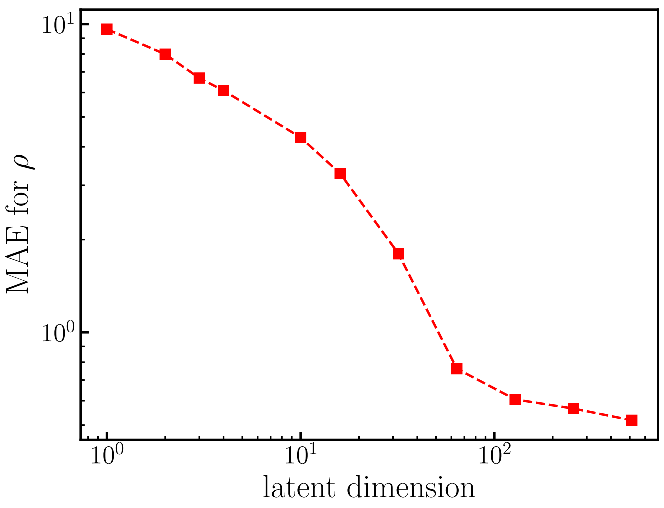

FIG. 5. The mean absolute errors (MAEs) for predicting density with the autoencoder as a function of the latent dimension d_z, taken as $d_z = 1, 2, 3, 4, 10, 2^4, 2^5, 2^6, 2^7, 2^8$, and $2^9 = 512$.

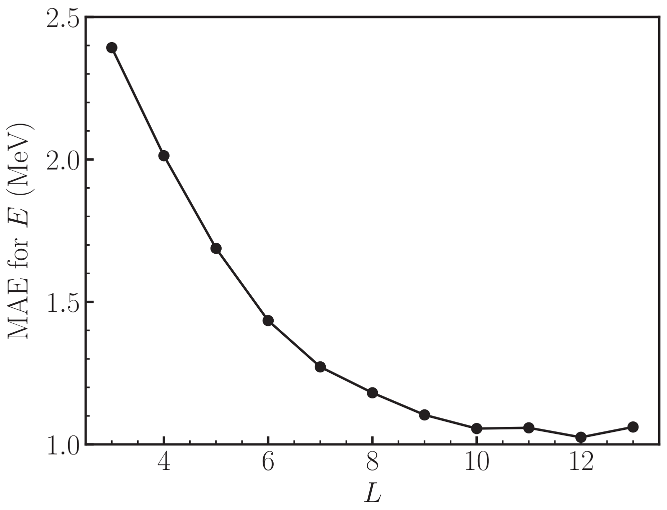

FIG. 6. The mean absolute errors (MAEs) for predicting energy from multipole moments. The vertical axis denotes the maximum multipoles in $S_L = \{Q_{20}, Q_{30}, \dots, Q_{L0}\}$, which is similar to the latent dimension in our MTL.

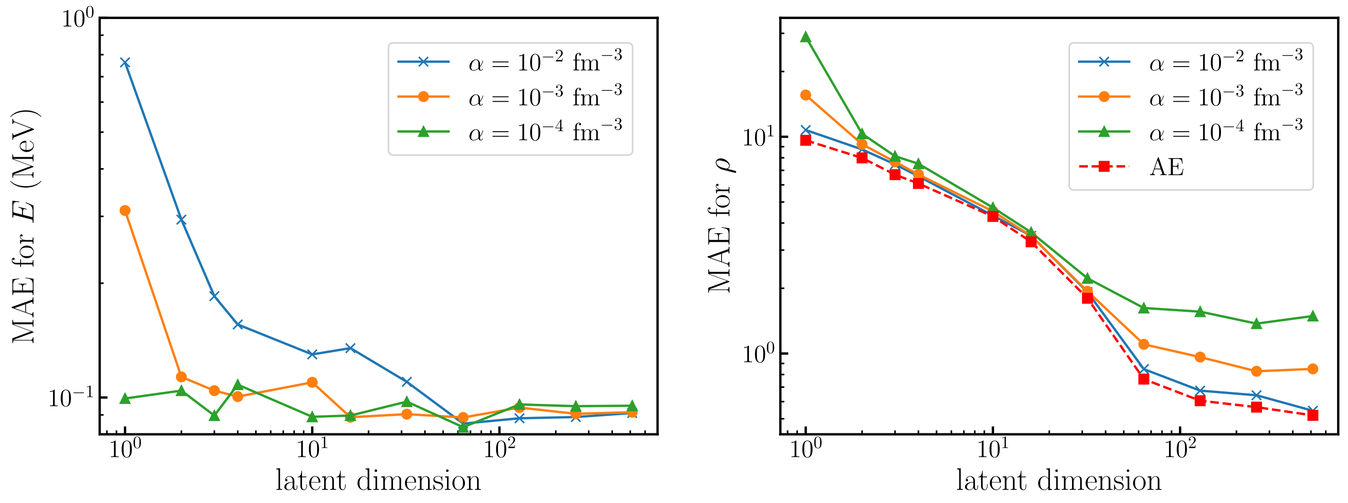

FIG. 7. The mean absolute errors (MAEs) of the densities and the energies as a function of the latent dimension for several values of the hyperparameter α in the MTL loss function given by Eq. (14). The dashed line on the right panel is the same as that plotted in Fig. 5, which shows the results of the autoencoders (AE).

that is, 64 keV. We also tried with a deeper model with more trainable parameters, but the results were not improved. As a reference, we compute the MAE by assuming that the output energies are given by uniform random numbers within a range of energies in our dataset, that is, the MAE obtained by replacing $E^{(i)}_{\text{pred}}$ in Eq. (17) with random numbers. The resultant value is 5.0 MeV. The errors with the low order multipole moments are reduced from this value only by a factor of about 2, indicating that the multipole moments do not contain enough information, at least for our dataset.

We also examine the density prediction from the multipole moments using the decoder $\mathcal{D}_\rho$. We find that the mean absolute errors are several times larger than the mass number $A = 236$, indicating that the multipole moments do not have a capacity to predict the densities, at least, for our dataset. Our results might be somewhat improved with techniques such as pretraining with another dataset, though. Rather than attempting it, however, in the next section we will propose using MTL to extract much more consistent variables for shape dynamics than the multipole moments.

IV. RESULTS OF MTL

A. Latent variables and dimensions

Let us now discuss the MTL results and the extracted latent variables. The MTL loss function has one hyperparameter α, which determines the relative priority between the errors of energy and those of density [see Eq. (14)]. We determine the value of α so that $\mathcal{L}_E$ and $\mathcal{L}_\rho$ are roughly on the same scale. In this study, we employ three different values for α, that is, $\alpha = 10^{-2}$, 10^{-3}, and 10^{-4} fm^{-3}.

Figure 7 illustrates the MAE for the energies (the left panel) and for the densities (the right panel). One can see that increasing the latent dimension rapidly increases the performance of the energy prediction, while the performance for the densities tends to increase only slowly. As the two competing tasks tend to increase the errors of each other, the density and energy errors behave oppositely when increasing or decreasing α. That is, while increasing α reduces the errors in the densities, it increases the errors in the energies.

Remarkably, even with a small number of the latent variables, the energy can be predicted within errors of at most a few hundred keV. In particular, when the latent dimension is increased from 1 to 2, the errors in energy decrease rapidly for $\alpha = 10^{-2}$ and 10^{-3} fm^{-3}. With two latent variables, the densities can also be predicted well: the density prediction errors for $d_z = 2$ are in fact larger than the results of the autoencoders (AE) only by a small amount (see the dashed line).

It is a good compromise to set $\alpha = 10^{-3}$ fm^{-3}. For larger values of α, the density prediction is better but the energy prediction is worse. On the other hand, for small values of α the density predication is worse. With $\alpha = 10^{-3}$ fm^{-3}, both the energy predication and the density prediction are simultaneously good. With this choice of α, the errors in energy do not decrease much when the latent dimension is increased from 2 to 3. This implies that the main part of the dynamics in our dataset can be characterized by only two parameters. We emphasize that our latent variables contain much more information on our dataset as compared to the conventional multipole moments. This can be regarded as evidence that the multipole moments may not provide appropriate collective coordinates to describe shape dynamics of heavy nuclei, such as ^{236}U, even in the region of small deviations from the ground state deformation.

In the cases where the latent dimension is smaller than 3, we can easily visualize the profile of the latent variables. Especially, the two- and three-dimensional cases have good performance compared with the criteria defined in Sec. III. Figure 8 shows scatter plots of the latent variables obtained with $d_z = 2$ (the left panel) and $d_z = 3$ (the right panel) together with $\alpha = 10^{-3}$ fm^{-3}. Notice that the latent variables do not have a physical dimension, and the scales in the figure are irrelevant. The color dimension denotes the binding energy

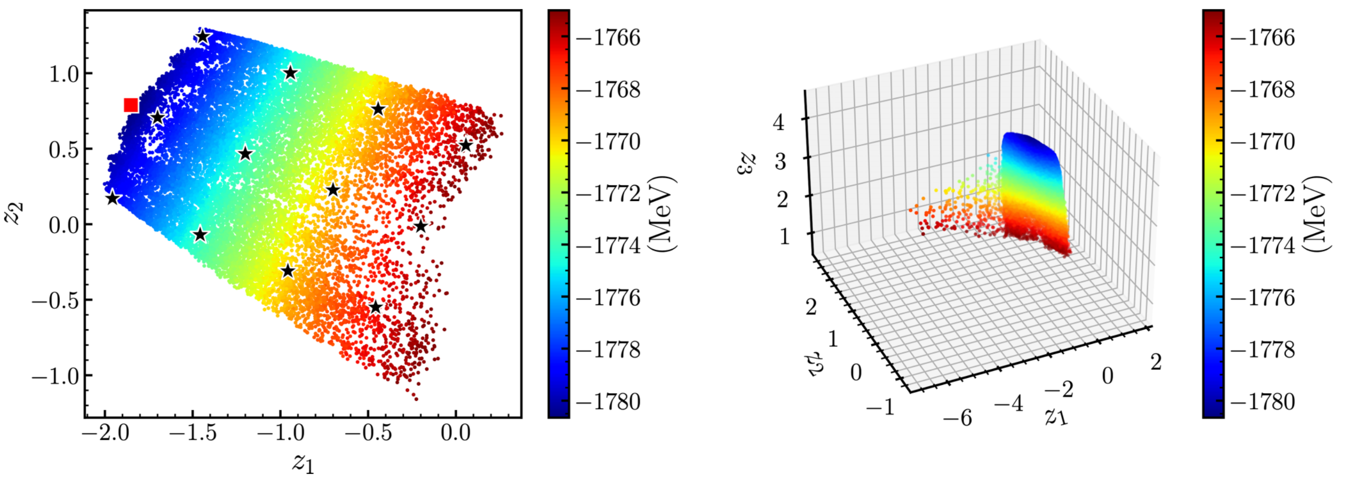

FIG. 8. The latent variables for a combination of the validation data (5%) and the test data (5%). The performances for these data are almost the same, and we include the validation data in this figure only for a presentation purpose. The latent dimensions are 2 (the left panel) and 3 (the right panel), and the hyperparameter of the MTL loss function is set to be $\alpha = 10^{-3}$ fm^{-3}. The binding energy of each data is denoted with color. In the left panel, the ground state with no external field is shown by a filled square, for which the corresponding latent variables are $(z_1, z_2) = (-1.85, 0.79)$. The black stars denote those points whose densities are reconstructed from the latent variables in Fig. 9.

of each data. Here, we plot both the validation data and the test data, since the errors for those data are almost the same. The ground state of ^{236}U is at $(z_1, z_2) = (-1.85, 0.79)$ if the external potential is not introduced, as shown by a square dot in the left panel. One can notice that there is a symmetric axis in the plots. To see the significance of the symmetric axis, the reconstructed densities are plotted in Fig. 9 for the points shown in the left panel of Fig. 8. Even though those densities contain certain errors as they are reconstructed ones, one can still see that the shapes on the opposite side across the symmetry axis can be regarded as the same shapes but inverted with respect to the z axis. This implies that our MTL model successfully recognizes the parity symmetry. Note that the direction of the symmetry axis and the position of the origin are not physically meaningful, as they are determined by the initial trainable parameters of the neural network. In fact, we performed exactly the same calculations but with another initial condition, and obtained a similar scatter plot with a different position of the origin of the z coordinate.

Let us focus on the case with $d_z = 2$. Figure 10 shows the quadruple moment and the octupole moment of each latent variable. One can see the existence of the symmetry axis. The figure also indicates that Q_{20} and Q_{30} do not play a special role in our dataset. An advantage of our latent variables is that all the points which correspond to excited states for given multipole moments can be treated as ground states at the corresponding latent variables. Even though it is trivial to obtain a smooth energy surface if the dimension of the latent variables is large, it is notable that this is achieved even with the dimension of 2. On the other hand, if the energies are plotted in the (Q_{20}, Q_{30}) plane, those energies are rather scattered, as one can see in Fig. 2. Note that all the configurations correspond to the ground state with external fields, but they do not necessarily correspond to the local ground state for each (Q_{20}, Q_{30}) in the constrained HFB.

To see how the multipole moments (Q_{20}, Q_{30}) in Fig. 2 map onto the two-dimensional plot for the latent variables shown in the left panel of Fig. 8, we carry out the constrained HFB calculations. Points connected with color arrows on the left panel of Fig. 11 correspond to the points connected with the same color on the right panel. One can see that the trajectories on the (z_1, z_2) plane are quite complex, and the mapping from (Q_{20}, Q_{30}) to (z_1, z_2) is highly nontrivial. Evidently, the main feature of our dataset captured by MTL cannot easily be described by conventional quadrupole moments and requires novel latent variables.

B. Domain shift

The fact that a neural network performs well on test data does not necessarily mean that the model will perform well when users try with their own data. This is because properties of two datasets may be different. To clarify this problem, we introduce the idea of a domain in the context of transfer learning (TL). Here, a domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$ consists of a feature (input) space $\mathcal{X}$ and a probability distribution $P(X)$, $X \in \mathcal{X}$ [53,54].[3] In this study, $\mathcal{X}$ corresponds to a set of densities as a whole, and $P(X)$ corresponds to the probability distribution of the densities $p[\rho]$ discussed in Sec. IIA. In machine learning, one usually prepares a labeled dataset for training and test by sampling from a source domain $\mathcal{D}^s$ and a target domain $\mathcal{D}^t$, respectively. In traditional machine learning methods, which we follow in this paper, $\mathcal{D}^s$ is assumed to be identical to $\mathcal{D}^t$. We have generated both the datasets by randomly splitting a single dataset, and therefore we have also adopted this assumption. However, the traditional machine learning methods do not guarantee good performance for a target domain when the domains are different, i.e., $\mathcal{D}^s \neq \mathcal{D}^t$. This problem of domain shift is quite crucial when one would like to adopt a learned model for actual applications. It is

[3]Introducing the idea of a task allows a more detailed and strict categorization of a problem. See Refs. [53,54] for details.

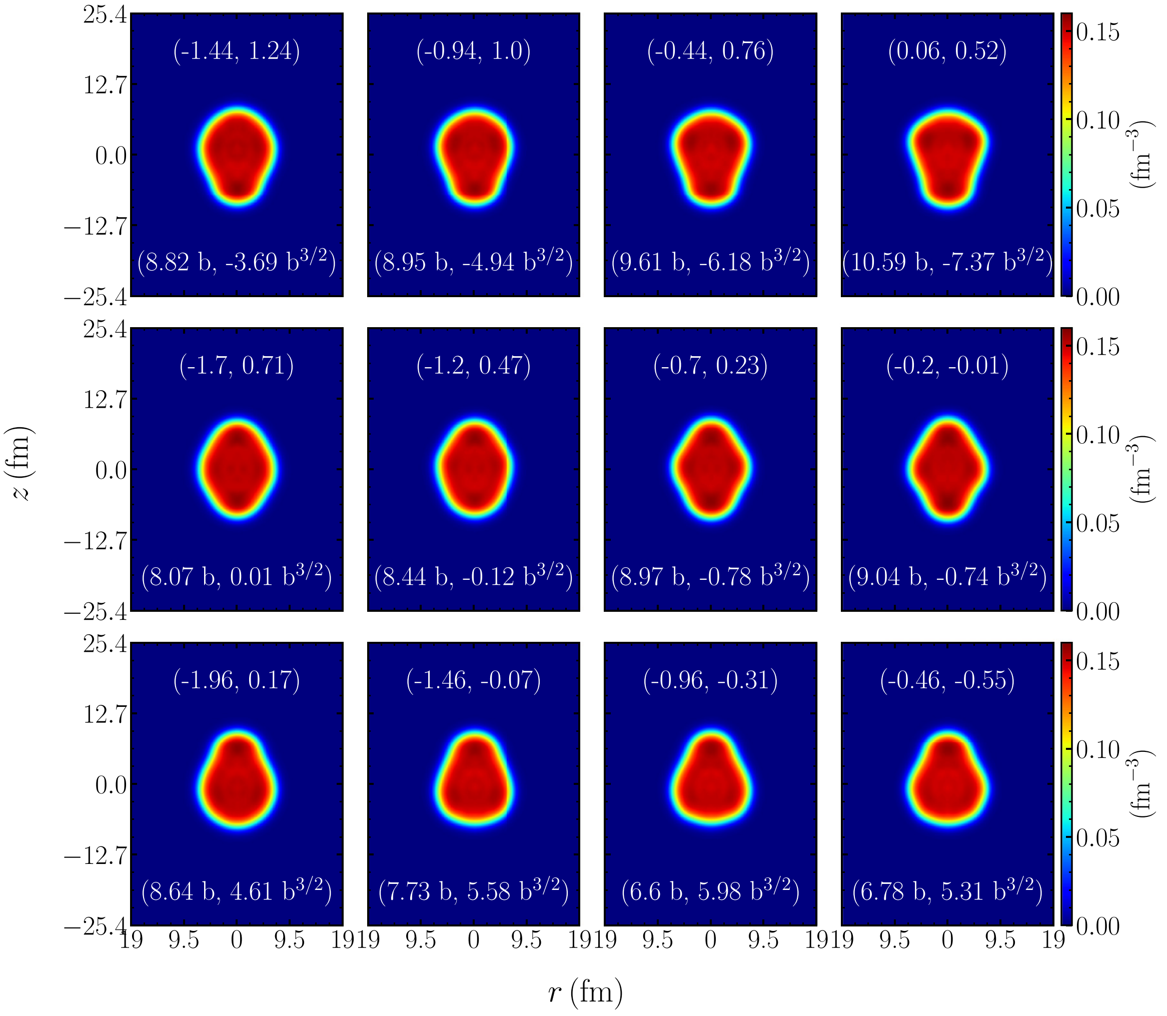

FIG. 9. Sample densities reconstructed from latent variables for the points shown in Fig. 8. The two numbers displayed on the top of each panel denote the latent variables (z_1, z_2), while those at the bottom are the quadrupole and the octupole moments, (Q_{20}, Q_{30}).

not uncommon that a performance of a model immediately declines when the model is actually applied, even when the model shows superior performance with data at hand.

In this subsection, we check the performance of our MTL model when it is applied to data generated by a domain which is different from the one of our dataset, namely, the source

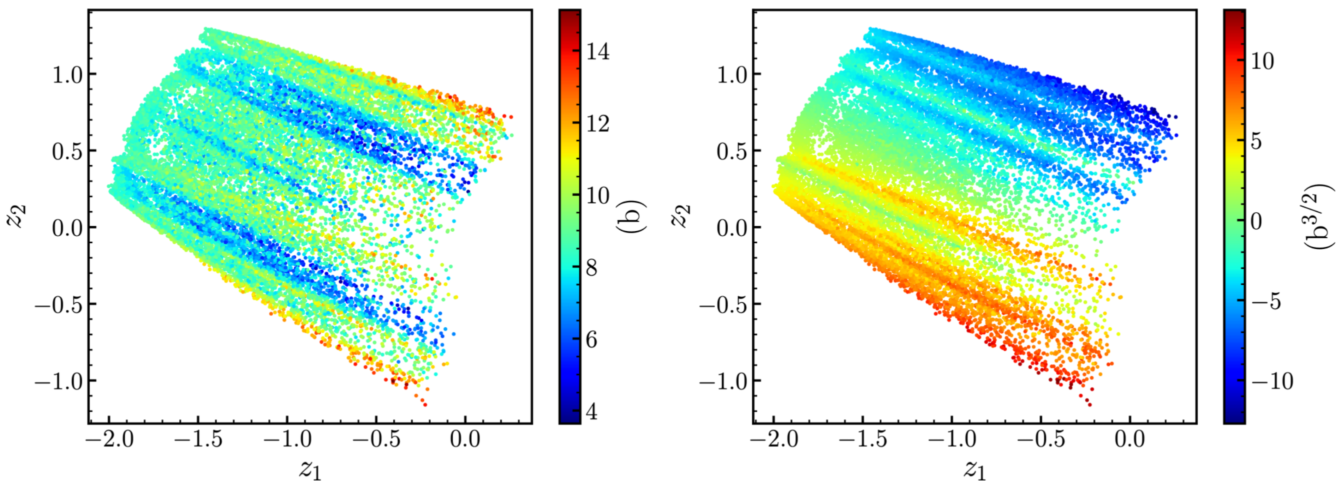

FIG. 10. Same as the left panel of Fig. 8, but for the quadrupole moment (the left panel) and the octupole moment (the right panel).

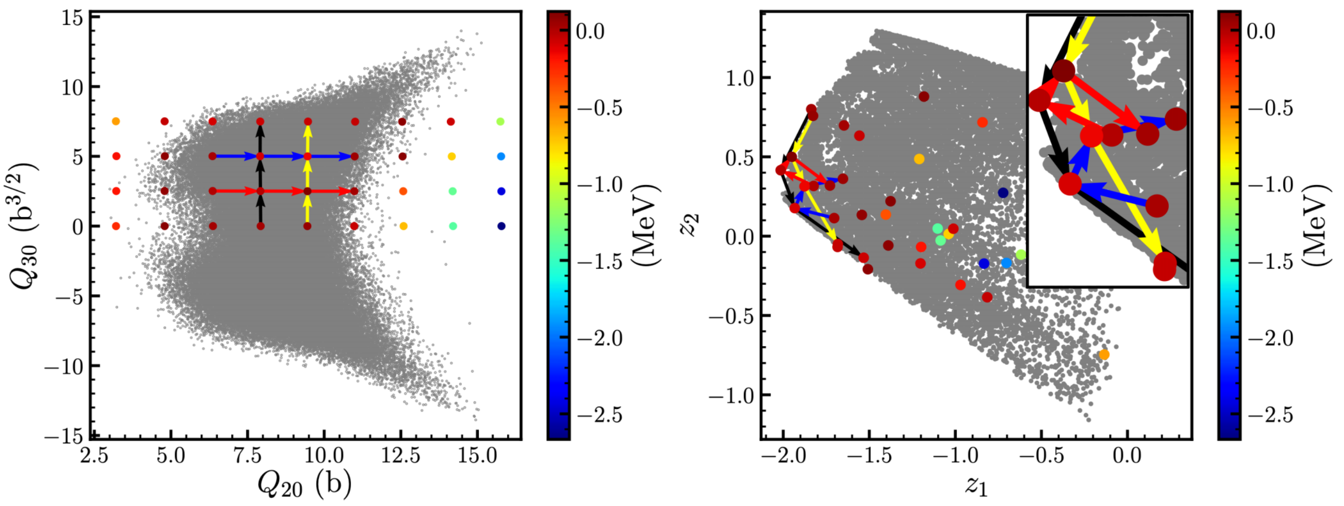

FIG. 11. Mapping from Fig. 2 to the left panel of Fig. 8. Points connecting with a color arrow on the left panel correspond to the points connecting with the same color in the right panel. The color for each points denotes the difference between the true energy and the predicted energy estimated with the MTL model with $\alpha = 10^{-3}$ fm^{-3} and $d_z = 2$.

domain. In nuclear physics, the constrained HFB method with Q_{20} and Q_{30} is often utilized, and we regard that as a target domain. Of course the corresponding dataset is expected to have some overlap with the dataset generated by our random potentials, but the probability would be quite small because the number of data is finite. Thus, we can consider that the target domain is different from the source domain in our dataset.

The colors of the points in Fig. 11 show the difference between the true energy and the predicted energy with the MTL model. In the left panel, one can see that the performance of our MTL model is high in the region where the training data exist in terms of multipole moments, indicating that our model is not overly adapted to the source domain. On the other hand, the performance becomes significantly worse outside the region.

For a fission problem, one would also need a calculation for large deformations which are outside the region studied in this paper. If we want the MTL model to have a prediction ability even for such larger deformations, we simply need to use larger datasets that include such deformed states. Alternatively, one can also use other techniques, such as domain adaptation (DA) [53,54] and domain generalization (DG) [69].

C. Symmetry and data augmentation

Even though the latent variables obtained in this study are good enough, one may obtain better latent variables by taking into account symmetry of the datasets. To check this, in this subsection we repeat MTL by constructing the datasets in which symmetry of the system is respected.

The densities constructed in this paper have axial symmetry along the z axis. Even though those densities in general break parity symmetry, the binding energy is invariant with respect to the parity operation. To reflect this physical symmetry in the learning results, it is desirable to have the same symmetry in the dataset as well. Since the source domain is invariant under the parity transformation, if one samples a quite large amount of data, one can assume that the distribution of the dataset should also be invariant under the parity transformation. However, this is difficult in practice with a finite amount of data. Therefore, we adopt the data augmentation of flipping images, for which the corresponding energy is exactly invariant due to the parity symmetry. In the actual calculations, we randomly flip an input image with respect to z axis with a probability of 0.5. In addition, we also introduce an improvement to the encoder of our MTL model. That is, as the original ResNets, which utilize odd kernel sizes even with a stride of 2, might cause a shift of a center of the image, we increase the kernel sizes by 1 for the convolutional layers when the stride is set to be 2.

We train the improved MTL model with the augmented dataset, with $d_z = 2, 3$ and $\alpha = 10^{-3}$ fm^{-3}. We evaluate the performance of the symmetry with the following two metrics:

$$M_E = \sum_{i=1}^{|D|} \frac{1}{|D|} \left| \mathcal{D}_E(\mathcal{E}[\rho^{(i)}]) - \mathcal{D}_E\left(\mathcal{E}\left[\rho_{\mathcal{F}}^{(i)}\right]\right) \right|, \tag{20}$$

$$M_\rho = \sum_{i=1}^{|D|} \frac{1}{|D|} \int d^3r \left| \mathcal{D}_\rho(\mathcal{E}[\rho^{(i)}]) - \mathcal{F}\left[\mathcal{D}_\rho\left(\mathcal{E}\left[\rho_{\mathcal{F}}^{(i)}\right]\right)\right] \right|, \tag{21}$$

where $\mathcal{F}$ is the flipping operation, and $\rho_{\mathcal{F}}$ is defined as $\mathcal{F}[\rho]$. If the model fully respected the symmetry, both of these metrics would be exactly zero.

Table II shows M_E and M_ρ for our datasets. One can see that the symmetry consideration does not significantly affect the performance of MTL. This is because our MTL models have enough representation ability, and the datasets are from the beginning symmetric enough to learn the symmetry. In fact, comparing the two metrics to the MAEs, the symmetry is correctly learned within the errors because the metrics are smaller. Rather, the influence of the initial values of the neural network is more significant. On the other hand, when we apply the data augmentation to supervised learning for the energy only with the improved ResNet18, we find that MAE

TABLE II. The metrics M_E and M_ρ defined by Eqs. (20) and (21) to evaluate the role of symmetry of the MTL models. These are evaluated only for test data (5%). For comparison, the MAEs for energy and density are also shown. If symmetry is False, the results for the MAEs are the same as those in Fig. 7. On the other hand, in the case of True, we apply the data augmentation as well as the improvement of the encoder.

Symmetry	d_z	M_E (MeV)	M_ρ	MAE E (MeV)	MAE ρ
False	2	0.0842	5.0618	0.1133	9.2352
True	2	0.0798	5.4921	0.1250	9.2756
False	3	0.0675	3.7661	0.1043	7.6681
True	3	0.0716	3.3035	0.1003	7.7055

is reduced to 52 keV. Therefore, we conclude that, while data augmentation itself is helpful in increasing accuracy, it does not necessarily contribute to the accuracy of models when they include adversarial tasks, such as our multitask learning.

V. SUMMARY AND FUTURE PERSPECTIVE

We have applied multitask learning (MTL) to a shape dynamics in the vicinity of the ground state of ^{236}U in order to extract a common feature representation of densities and binding energies. To this end, we have employed the Skyrme Kohn-Sham density functional theory (Skyrme KS-DFT) with random external potentials. In our MTL models, the input densities are compressed into latent variables by an encoder, and they are reconverted into energies and densities by decoders. The resultant latent variables can be regarded as a kind of collective coordinates, as the manifold hypothesis and an assumption of collective coordinates are conceptually close to each other. We have shown that latent variables with dimension of as small as 2 well reproduce the energies and the densities of the test data. On the other hand, we have shown that conventional multipole moments contain much less information on shape dynamics in our dataset. This suggests that it is important to choose collective coordinates by appropriately taking into account dynamics, as in our latent variables.

In this study we have utilized the autoencoder only for an analysis of latent variables. When one would like to apply our idea as a generative model, one should adopt probabilistic models, such as a variational autoencoder (VAE) [39,70]. Notice that, in an image generation, it is difficult to obtain generalization performance and generate novel images which are not included in a training dataset. To handle this, generative adversarial networks (GANs) [71,72] and, in the latest studies, diffusion models (DMs) [73–75] are gaining popularity in this field. In particular, the DMs have achieved state-of-the-art performance for text to image generation tasks. Therefore, one may think of applying the latest probabilistic models to nuclear physics.

To extract latent variables that consistently describe nuclear fission, it would be necessary to collect a large number of data on larger deformed states. In that case, the dimension of latent variables may increase, even though the dimension of 2 is sufficient for shape dynamics near the ground state. However, we found it difficult to obtain such data with random potentials, especially data at unstable locations such as a barrier top. To obtain such data, one would need either a huge amount of effort or a drastic improvement of the numerical algorithm for convergence of the DFT. In addition, since the calculations have been performed in the framework of density functional theory, no information on the many-body wave functions are available from our MTL, and thus our method cannot be connected to GCM or other methods that deal with many-body wave functions. Clearly, novel ideas are necessary in order to connect the latent variables to collective models so that, e.g., the moment of inertia can be computed.

One of the important conclusions in this paper is that sufficiently large data sets themselves contain a wealth of information on the dynamics. If one analyzes them in an appropriate manner, one can immediately extract information on the dynamics. Especially in the context of KS-DFT, one can avoid using phenomenological constraining fields, which are only convenient mathematical tools in most cases in nuclear physics. This merit enables one to recast nuclear physics from a new perspective.

Fortunately, due to the remarkable growth of the machine learning field in recent years, there have been many useful tools to analyze big data. Unfortunately, however, there is still a lack of datasets to learn in nuclear theory. Therefore, at this stage, the generation of high quality datasets such as ImageNet [60] is still awaited. For such data,vision transformer (ViT) [67] is expected to perform better than the conventional neural network models. In addition, it is also difficult at this moment to train models with more than a billion parameters. This is because CPUs have been mainly utilized in nuclear physics and it may still be difficult to access to a large number of expensive GPUs. Moreover, it is also important to secure GPUs, which will be even in a greater demand in future.

ACKNOWLEDGMENTS

We thank A. Sannai for useful advice on multitask learning. We also thank K. Yoshida for discussions. A part of numerical computations in this work was carried out at the Yukawa Institute Computer Facility. This work was supported by JSPS KAKENHI Grants No. JP22KJ1697, No. JP19K03861, and No. JP23K03414.

[1] A. Bulgac, P. Magierski, K. J. Roche, and I. Stetcu, Phys. Rev. Lett. **116**, 122504 (2016).

[2] N. Schunck and D. Regnier, Prog. Part. Nucl. Phys. **125**, 103963 (2022).

[3] M. Brack, J. Damgaard, A. S. Jensen, H. C. Pauli, V. M. Strutinsky, and C. Y. Wong, Rev. Mod. Phys. **44**, 320 (1972).
[4] N. Schunck and L. M. Robledo, Rep. Prog. Phys. **79**, 116301 (2016).
[5] J. Sadhukhan, S. A. Giuliani, and W. Nazarewicz, Phys. Rev. C **105**, 014619 (2022).
[6] E. Flynn, D. Lay, S. Agbemava, P. Giuliani, K. Godbey, W. Nazarewicz, and J. Sadhukhan, Phys. Rev. C **105**, 054302 (2022).
[7] A. Staszczak, A. Baran, J. Dobaczewski, and W. Nazarewicz, Phys. Rev. C **80**, 014309 (2009).
[8] K. Washiyama, N. Hinohara, and T. Nakatsukasa, Phys. Rev. C **103**, 014306 (2021).
[9] Y. Aritomo and S. Chiba, Phys. Rev. C **88**, 044614 (2013).
[10] C. Ishizuka, M. D. Usang, F. A. Ivanyuk, J. A. Maruhn, K. Nishio, and S. Chiba, Phys. Rev. C **96**, 064616 (2017).
[11] J. Randrup and P. Möller, Phys. Rev. Lett. **106**, 132503 (2011).
[12] J. Randrup, P. Möller, and A. J. Sierk, Phys. Rev. C **84**, 034613 (2011).
[13] H. Goutte, J. F. Berger, P. Casoli, and D. Gogny, Phys. Rev. C **71**, 024316 (2005).
[14] D. Regnier, N. Dubray, and N. Schunck, Phys. Rev. C **99**, 024611 (2019).
[15] J. Zhao, T. Nikšić, and D. Vretenar, Phys. Rev. C **105**, 054604 (2022).
[16] T. Marumori, T. Maskawa, F. Sakata, and A. Kuriyama, Prog. Theor. Phys. **64**, 1294 (1980).
[17] M. Matsuo, Prog. Theor. Phys. **76**, 372 (1986).
[18] M. Matsuo, T. Nakatsukasa, and K. Matsuyanagi, Prog. Theor. Phys. **103**, 959 (2000).
[19] N. Hinohara, T. Nakatsukasa, M. Matsuo, and K. Matsuyanagi, Prog. Theor. Phys. **119**, 59 (2008).
[20] K. Wen and T. Nakatsukasa, Phys. Rev. C **96**, 014610 (2017).
[21] R.-D. Lasseri, D. Regnier, J.-P. Ebran, and A. Penon, Phys. Rev. Lett. **124**, 162502 (2020).
[22] G. Saxena, A. Jain, and P. Sharma, Phys. Scr. **96**, 125304 (2021).
[23] Z.-A. Wang and J. Pei, Phys. Rev. C **104**, 064608 (2021).
[24] X. Wu, L. Guo, and P. Zhao, Phys. Lett. B **819**, 136387 (2021).
[25] X. Wu, Y. Lu, and P. Zhao, Phys. Lett. B **834**, 137394 (2022).
[26] M. R. Mumpower, T. M. Sprouse, A. E. Lovell, and A. T. Mohan, Phys. Rev. C **106**, L021301 (2022).
[27] A. E. Lovell, A. T. Mohan, T. M. Sprouse, and M. R. Mumpower, Phys. Rev. C **106**, 014305 (2022).
[28] O. M. Molchanov, K. D. Launey, A. Mercenne, G. H. Sargsyan, T. Dytrych, and J. P. Draayer, Phys. Rev. C **105**, 034306 (2022).
[29] X. H. Wu, Z. X. Ren, and P. W. Zhao, Phys. Rev. C **105**, L031303 (2022).
[30] M. Knöll, T. Wolfgruber, M. L. Agel, C. Wenz, and R. Roth, Phys. Lett. B **839**, 137781 (2023).
[31] X. Zhang, W. Lin, J. M. Yao, C. F. Jiao, A. M. Romero, T. R. Rodríguez, and H. Hergert, Phys. Rev. C **107**, 024304 (2023).
[32] G. P. A. Nobre, D. A. Brown, S. J. Hollick, S. Scoville, and P. Rodríguez, Phys. Rev. C **107**, 034612 (2023).
[33] Z.-X. Yang, X.-H. Fan, Z.-P. Li, and H. Liang, Phys. Lett. B **840**, 137870 (2023).
[34] A. Boehnlein, M. Diefenthaler, N. Sato, M. Schram, V. Ziegler, C. Fanelli, M. Hjorth-Jensen, T. Horn, M. P. Kuchera, D. Lee, W. Nazarewicz, P. Ostroumov, K. Orginos, A. Poon, X.-N. Wang, A. Scheinker, M. S. Smith, and L.-G. Pang, Rev. Mod. Phys. **94**, 031003 (2022).
[35] X. H. Wu and P. W. Zhao, Phys. Rev. C **101**, 051301(R) (2020).
[36] M. Usama, J. Qadir, A. Raza, H. Arif, K.-l. A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, IEEE Access **7**, 65579 (2019).
[37] S. S. Mousavi, M. Schukat, and E. Howley, in *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016*, edited by Y. Bi, S. Kapoor, and R. Bhatia (Springer International, Cham, 2018), pp. 426–440.
[38] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, IEEE Trans. Knowl. Data Eng. **35**, 857 (2023).
[39] M. Tschannen, O. Bachem, and M. Lucic, arXiv:1812.05069.
[40] C. Fefferman, S. Mitter, and H. Narayanan, J. Amer. Math. Soc. **29**, 983 (2016).
[41] M. Verriere, N. Schunck, I. Kim, P. Marević, K. Quinlan, M. N. Ngo, D. Regnier, and R. D. Lasseri, Front. Phys. **10**, 1028370 (2022).
[42] R.-D. Lasseri, D. Regnier, M. Frosini, M. Verriere, and N. Shunck, arXiv:2306.08348.
[43] M. Crawshaw, arXiv:2009.09796.
[44] W. Kohn and L. J. Sham, Phys. Rev. **140**, A1133 (1965).
[45] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964).
[46] M. Bender, P.-H. Heenen, and P.-G. Reinhard, Rev. Mod. Phys. **75**, 121 (2003).
[47] D. Vautherin and D. M. Brink, Phys. Rev. C **5**, 626 (1972).
[48] P. Ring and P. Schuck, *The Nuclear Many-Body Problem* (Springer-Verlag, New York, 1980).
[49] R. N. Perez, N. Schunck, R.-D. Lasseri, C. Zhang, and J. Sarich, Comput. Phys. Commun. **220**, 363 (2017).
[50] E. Chabanat, P. Bonche, P. Haensel, J. Meyer, and R. Schaeffer, Nucl. Phys. A **635**, 231 (1998).
[51] A. S. Umar and V. E. Oberacker, Phys. Rev. C **74**, 021601(R) (2006).
[52] K. Mills, M. Spanner, and I. Tamblyn, Phys. Rev. A **96**, 042113 (2017).
[53] M. Wang and W. Deng, Neurocomputing **312**, 135 (2018).
[54] G. Csurka, arXiv:1702.05374.
[55] K. Ryczko, D. A. Strubbe, and I. Tamblyn, Phys. Rev. A **100**, 022512 (2019).
[56] N. Hizawa, K. Hagino, and K. Yoshida, Phys. Rev. C **108**, 034311 (2023).
[57] F. Brockherde, L. Vogt, L. Li, M. E. Tuckerman, K. Burke, and K.-R. Müller, Nat. Commun. **8**, 872 (2017).
[58] K. He, X. Zhang, S. Ren, and J. Sun, in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '16* (IEEE Computer Society, Los Alamitos, CA, 2016), pp. 770–778.
[59] LSVRC2015, https://www.image-net.org/challenges/LSVRC/2015/index.php.
[60] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, in *2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE Computer Society, Los Alamitos, CA, 2009), pp. 248–255.
[61] K. He, X. Zhang, S. Ren, and J. Sun, in *2015 IEEE International Conference on Computer Vision (ICCV)* (IEEE Computer Society, Los Alamitos, CA, 2015), pp. 1026–1034.
[62] A. Odena, V. Dumoulin, and C. Olah, Distill (2016), doi:10.23915/distill.00003.
[63] R. Cipolla, Y. Gal, and A. Kendall, in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Los Alamitos, CA, USA, 2018), pp. 7482–7491.

[64] S. Liu, E. Johns, and A. J. Davison, in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Los Alamitos, CA, USA, 2019), pp. 1871–1880.
[65] D. P. Kingma and J. Ba, arXiv:1412.6980.
[66] Pytorch, https://pytorch.org.
[67] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, arXiv:2010.11929.
[68] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, in *Advances in Neural Information Processing Systems*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Red Hook, New York, 2017), Vol. 30.
[69] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, IEEE Trans. Pattern Anal. Mach. Intell. **45**, 4396 (2023).
[70] D. P. Kingma and M. Welling, arXiv:1312.6114.
[71] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, in *Advances in Neural Information Processing Systems*, Vol. 27, edited by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (Curran Associates, Red Hook, New York, 2014).
[72] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, IEEE Trans. Knowl. Data Eng. **35**, 3313 (2023).
[73] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, in *Proceedings of the 32nd International Conference on Machine Learning*, edited by F. Bach and D. Blei, Proceedings of Machine Learning Research, Vol. 37 (PMLR, Lille, France, 2015), pp. 2256–2265.
[74] J. Ho, A. Jain, and P. Abbeel, in *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Curran Associates, 2020), Vol. 33, pp. 6840–6851.
[75] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang, arXiv:2209.00796.