Calibration of nuclear charge density distribution by back-propagation neural networks

Zu-Xing Yang (杨祖星),1 Xiao-Hua Fan (范小华),2 Tomoya Naito (内藤智也),3,4 Zhong-Ming Niu (牛中明),5 Zhi-Pan Li (李志攀),2 and Haozhao Liang (梁豪兆)4,3

1RIKEN Nishina Center, Wako 351-0198, Japan
2School of Physical Science and Technology, Southwest University, Chongqing 400715, China
3RIKEN Interdisciplinary Theoretical and Mathematical Sciences Program, Wako 351-0198, Japan
4Department of Physics, Graduate School of Science, The University of Tokyo, Tokyo 113-0033, Japan
5School of Physics and Optoelectronic Engineering, Anhui University, Hefei 230601, China

Based on the back-propagation neural networks and density functional theory, a supervised learning is performed firstly to generate the nuclear charge density distributions. The charge density is further calibrated to the experimental charge radii by a composite loss function. It is found that, when the parity, pairing, and shell effects are taken into account, about 96% of the nuclei in the validation set fall within two standard deviations of the predicted charge radii. The calibrated charge density is then mapped to the matter density, and further mapped to the binding energies according to the Hohenberg-Kohn theorem. It provides an improved description of some nuclei in both binding energies and charge radii. Moreover, the anomalous overbinding in $^{48}$Ca implies the existence of an indispensable beyond-mean-field effect.

Introduction—The charge density distribution, experimentally given by electron scattering experiments [1–3], is essential for extracting nuclear structure information, including the shell-structure evolution, shape co-existence, shape transition, and neutron-skin thickness [4–7]. The relative radii of neighboring nuclei can also be determined using muonic-atom spectra as well as isotope shifts of laser spectroscopy [1]. Since a strong connection exists between charge densities and matter densities, one generally calculates the charge density distribution by merging the nucleon form factors [8, 9] with the matter densities determined by scattering experiments, the $(−1p)$ reaction [10], or even heavy-ion collisions [11].

Early on, the Fermi distributions [12, 13] and Fourier-Bessel expansion [13, 14] were used to describe nuclear matter and charge densities approximately. With the development of computational power, various branching models [15–24] derived from density functional theory (DFT) and shell model calculation have become more popular among theorists. However, due to the complexity of nuclear many-body systems, the calculations of these theories still face challenges in describing the beyond-mean-field effects and nucleon-nucleon correlations [25–28].

For nuclear complex systems, back-propagation neural networks [29] have achieved a series of success in various aspects, such as nuclear masses [30–32], nuclear spins and parities [33], charge radii [34–37], excited states [38, 39], extrapolation problems in an ab initio method [40], α-decay half-lives [41], β-decay half-lives [42], fission yields [43, 44], and so on. Most present neural networks learn and predict the residuals between theoretical and experimental values, which has an advantage that the corrected predictions are more accurate than the existing theoretical model [30, 42, 45]. However, the corrections have weak physical interpretability and it is difficult to predict different observables consistently. Google proposed a hybrid quantum-classical machine learning model for training beyond classical data types, where back-propagation is used to tune the quantum logic gate parameters, allowing for deep integration of physics and neural networks [46].

Based on the Hohenberg-Kohn maps [47] of DFT and the proven strong generalization ability of neural networks in describing density distributions [48], in this letter, we collectively constrain the charge density distributions by back-propagation with experimental data of charge radii, which makes the residual information flow back from radii to densities. We also construct the map from charge density to binding energy to achieve further transfer of information from radius to binding energy.

The charge density generators—Previously, a multilayer feed-forward neural network with a back-propagation algorithm of error has been elaborated to perform the maximum likelihood estimation in the process of generating density distributions [48], approximating the theoretical calculation. It is shown that a network trained by the density distributions of about 200–300 nuclei is sufficient to describe the density distributions of all the nuclei on the nuclear chart and has a powerful extrapolation capability [48].

Based on such a high computational efficiency and generalization ability, in this study we introduce the correlation between the theoretical charge density and the experimental root-mean-square charge radius ($R_c$) to a new hybrid neural network. The structure of this network is shown in Fig. 1, where $ρ_{c,i} = ρ_c(r_i)$ is the density on the mesh with $r_i = 0.1 \times i$ fm ($i = 0, 1, \ldots, 149$). The input is the proton and neutron numbers of a nuclide, $x = \{Z, N\}$. The outputs are the charge density...
The process of machine learning is divided into two stages: simulating the charge density distribution \( \rho_{c,\text{pre}} \) and root-mean-square charge radius \( R_c \) gained by the customized integration layer \( \hat{I}_{\rho_c \rightarrow R_c} \),

\[
\hat{I}_{\rho_c \rightarrow R_c} : R_{c,\text{pre}} = \sqrt{\frac{\int \rho_{c,\text{pre}} r^2 dr}{\int \rho_{c,\text{pre}}^2 dr}}, \tag{1}
\]

where \( \rho_{c,\text{pre}} \) is the normalized predicted charge density distribution, i.e., \( \int 4\pi \rho_{c,\text{pre}}(r)r^2 dr = Z \). For the hidden module, three attempts are made: 1. A deep fully connected neural network; 2. A convolutional neural network, suitable for learning the gradient information of features; 3. A neural network with a feature layer \( i_{12 \rightarrow i_{18}} \), containing the parity, paring, and shell effects,

\[
i_{12 \rightarrow i_{18}} : \{Z, N\} \rightarrow \{Z, N, O_p, O_n, \delta, v_p, v_n, P\} \tag{2}
\]

where

\[
O_p = Z \mod 2, \quad O_n = N \mod 2, \quad \delta = [(-1)^Z + (-1)^N]/2, \quad P = v_p v_n/(v_p + v_n), \tag{3}
\]

and \( v_p (v_n) \) is the difference between the actual nucleon number \( Z (N) \) and the nearest magic number (8, 20, 28, 50, 82, or 126). Hereafter, we refer to the three schemes of neural networks \( M \) as charge density generator-1 (CDG-1), CDG-2, and CDG-3, respectively. See Supplemental Materials (SM) for details. The map is denoted as

\[
(\rho_c, R_c)_{\text{pre}} = M(x, w), \tag{4}
\]

where \( w \) is the set of trainable parameters.

In general, the charge density distribution calculated by physical models \( \rho_{c,\text{theo}} \), e.g., that calculated by the Skyrme Hartree-Fock (SHF) theory with the Bardeen-Cooper-Schrieffer (BCS) pairing, is expected to be quite accurate. The \( R_c \) residuals between theory and experiment can be eliminated by assuming a correction to the charge density \( \delta \rho_c \), satisfying \( \int_0^\infty \delta \rho_c (r) r^2 dr = 0 \). We refer this process as calibration. The calibrated charge density distribution \( \rho_{c,\text{cali}} \), whose \( R_c \) is expected to be close to \( R_{c,\text{exp}} \) [1], can be obtained by

\[
\rho_{c,\text{cali}} (r) = \rho_{c,\text{theo}} (r) + \delta \rho_c (r). \tag{5}
\]

Additionally, we aim at making the smallest possible corrections \( \delta \rho_c \) to the theory.

To this end, we design a composite loss function. The normalized mean-square-error \( L_p \) [48] is employed as an assessment of density distribution:

\[
L_p = \frac{1}{N_g} \sum_{i=0}^{149} \left[ \rho_{c,\text{pre}} (r_i) - \rho_{c,\text{theo}} (r_i) \right]^2 \times 1 \text{fm}^6, \tag{6}
\]

where \( N_g = 150 \) indicates the number of grid points and the factor \( 1 \text{fm}^6 \) makes \( L_p \) dimensionless. The charge density distributions \( \rho_{c,\text{theo}}(r) \) are calculated by the SHF+BCS theory with the SkM⁺ interaction [49]. They are obtained from the charge form factor \( F_c \) by the inverse Fourier-Bessel transform,

\[
\rho_{c,\text{theo}} (r) = \frac{1}{2\pi^2} \int dk k^2 j_0 (kr) F_c (k; \rho_n, \rho_p, ...). \tag{7}
\]

where \( j_0 \) is the spherical Bessel function. The contributions of matter density and spin-orbit current are folded in \( F_c \) [50] (see SM for details). \( R_c \) has a large range of variation (about 1–6 fm) and therefore the Pearson \( \chi^2 \) divergence is picked, i.e., its loss function \( L_R \) reads

\[
L_R = \frac{(R_{c,\text{pre}} - R_{c,\text{exp}})^2}{R_{c,\text{pre}}} \times 1 \text{fm}^{-1}, \tag{8}
\]

where the factor \( 1 \text{fm}^{-1} \) also makes \( L_R \) dimensionless. The two loss functions \( L_p \) and \( L_R \) are combined as

\[
\text{Loss}(W, w) = \frac{1}{B_s} \sum_{n_s=1}^{B_s} [(1 - W) L_{p^{(n_s)}} (w) + W L_{R^{(n_s)}} (w)], \tag{9}
\]

where \( W \) is the weighting factor and \( B_s = 64 \) is batch size, which means that 64 nuclei are constrained simultaneously for each training session. Actually, the corrections for different nuclei are derived from the same parameter updates \( \delta w \), i.e., the correction of each nucleus is uniformly constrained by the other nuclei. We randomly take 640 nuclei (10 batches) of about 900 nuclei measured to date as the training set. The remaining nuclei are recorded as the validation set (see SM for details). It is recorded as an epoch when all nuclei on the training set have been trained once.

**Machine learning processes**—The process of machine learning is divided into two stages: simulating the
SHF+BCS results and correcting with experimental data. The evaluation of these processes is shown in Fig. 2. The first 3000 epochs are the stage of simulating the SHF+BCS results, during which \( \text{Loss}(0, w) = L_p \) is being minimized. One can see that the loss functions of the training and validation sets almost overlap. It means neither \( \rho \) nor \( R_c \) is overfitted, which shows the generalization ability of the network. The dashed magenta line is the Pearson \( \chi^2 \) divergence \( L_{R, \text{val}} \) on the validation set between the SHF+BCS and experimental values:

\[
L_{R, \text{theo}} = \frac{1}{N_v} \sum_{n_v \in \text{val}} (\frac{R_{c, \text{theo}}^{(n_v)}}{R_{c, \text{theo}}} \frac{R_{c, \text{exp}}^{(n_v)}}{R_{c, \text{exp}}} - 1)^2 \times 1 \text{ fm}^{-1},
\]

where \( N_v \) is the number of nuclei on the validation set. After a short training, the loss values \( L_R \) overlap with \( L_{R, \text{theo}} \). This indicates that the network naturally captures the \( R_c \) information well in the process of learning the density distribution.

After 3000 epochs, the pre-trained model is further tuned with an objective function \( \text{Loss}(W = 0.7) \), which allows the importance of experimental data to slightly exceed that of theoretical calculations. Thus, the charge density distributions under constraints are spontaneously corrected by the network. Since the correction makes the predicted charge radius close to the experimental value and the distribution naturally deviate from the model, the value of \( L_R \) falls and the value of \( L_p \) jumps as shown in Fig. 2. It is clear that machine learning successfully eliminated a portion of residuals between SHF+BCS and experimental values. Such a process can be migrated to any other theoretical model. Remarkably, the training costs only 10 GPU minutes.

**Calibrated charge density and charge radius**—To explore the network performance, the errors of the densities and radii given by the CDG-1, CDG-2, and CDG-3 on the validation set are plotted in the inserts of Fig. 2. It can be seen that the errors by CDG-3 are minimal for both density and radius. This indicates the predictions of CDG-3 are closest to the experimental radii, while its corrections to the theory are smallest. This agrees with our assumption. Compared to CDG-1 and CDG-2, CDG-3 takes into account the odd-even staggering, pairing, and shell effects. Therefore, we conclude that the network structures considering more physical properties are more sufficient to improve the prediction accuracy, which is consistent with the mass research [30].

Accuracy is more intuitive to show the CDGs’ performance than error. Figure 3(a) shows the prediction accuracy of the three networks, where the accuracy indicates the percentages of the experimental data on the validation set falling within different predicted standard deviations (red for 1 \( \text{std.} \), blue for 1.5 \( \text{std.} \), and pink for 2 \( \text{std.} \)). (b) Charge radii predicted by CDG-3, where the nuclei in the training set are denoted with the blue squares.
CDG-2. Meanwhile, the accuracy of CDG-3 (1 std.) is already close to 80%, which even reaches 96% for 2 std. It can be speculated that CDG-3 can predict the radii of the remaining ~ 2000 unmeasured nuclei with high precision. We present the predictions for nuclear charge radii in Fig. 3(b).

By taking several isotopic chains as examples, we compare the charge radii calculated by SHF+BCS and the predictions of CDG-3 with 1-std. error with the available experimental data in Fig. 4. It is found that CDG-3, containing the parity, paring, and shell effects, describes well the odd-even staggering for the Ca isotopes. Meanwhile, CDG-1 and CDG-2 failed in benchmarking the experimental data (see SM). For Pb isotopes, CDG-3 performs comparably with the theoretical model. The present predictions almost match the experimental data in the case of training several Pb isotopes. In particular, the predictive power remains strong even when we remove the Ca and Pb isotopes from the training set (see SM). Given the excellent performance of CDG-3, we predicted the untrained Dy and Tb isotopes. For the Dy isotopes, the predictions are in agreement with experiment, but with much smaller uncertainty than the experimental error. For the Tb isotopes, compared to the SHF+BCS results, the predicted isotope shifts are in good agreement with the experimental data, although the predicted charge radii are systematically larger than the present experimental central values by ~ 0.1 fm. The above predictions can be testified in the coming experiments.

Map from charge density to binding energy—According to the Hohenberg-Kohn theorem [47, 51], there exists a bijective map between the local matter density and the one-body potential. This means that the corrections on the density distributions can be mapped to other observables, such as binding energies [52]. Thus, we reconstruct a map from charge density to binding energy (CDTBE). The schematic structure of such a feedforward neural network is shown in Fig. 5. There are three parts in the CDTBE map—M1, M2, and M3. M1 is the previously trained charge density generator, and we take the CDG-3; M2 is a map from charge density to matter density as the inverse of Eq. (7); M3 is a map from the neutron, proton, and charge densities to the binding energy per nucleon. M2 and M3 are trained by the densities and binding energies of SHF+BCS with effective interactions SkM, SkM*, SkIII, SLy4, SkT, and SkT3 [53] (See SM for details). It is found that the networks adequately captures the relationship between densities and binding energy with different DFT effective interactions. As the charge density updates in CDG-3, the residual information flows to other observables, i.e., \( \delta R_c \rightarrow \delta \rho_c \rightarrow \{ \delta \rho_n, \delta \rho_p \} \rightarrow E/A \).

Taking the Ca isotopes as examples, the binding energies per nucleon obtained by the CDG-3 residual information flow are shown in Fig. 6. We note that the propagated corrections provide a better description, except \(^{47,48,49}\)Ca. Especially for \(^{40,42,44}\)Ca, both radii (see Fig. 4(a)) and binding energies coincide well with the experimental values. This illustrates that the physical features contained in the CDTBE map based on density functional theory are adequate for these nuclei. In contrast, for \(^{48}\)Ca, the radius is corrected to a smaller value than that of SHF+BCS, leading to an increase of nuclear densities in the central region and a further decrease of binding energy, which is consistent with equation of state. However, it results in further deviations from the experimental value, which implies an dispens-
able beyond-mean-field effect near $^{48}$Ca, as discussed in Ref. [54].

Summary—In this study, a novel supervised learning on the combination of the theoretical charge density distributions and the experimental charge radii has been performed. In such a way, the physics features embedded in nuclear density functional theory can be preserved to a large extent and the feedback from the experimental data can be considered quantitatively. It is found that the description of charge radii can be improved globally on the nuclear chart. In particular, the specific charge radii evolution in the Ca isotopes can be well reproduced by taking the parity, pairing, and shell effects into account. This property remains valid even when all the Ca isotopes are excluded in the learning set. The predictive power is also shown with the charge radii in the Dy and Tb isotopes, where the present experimental uncertainties are much larger than the prediction uncertainties.

Inspired by the Hohenberg-Kohn theorem, an information flow from charge density to binding energy has also been constructed and investigated. It is found that for the Ca isotopes the improvement in the description of charge radii can also be propagated to the improvement in the description of binding energies, except $^{47,48,49}$Ca. The corresponding analysis implies the existence of an indispensable beyond-mean-field effect near $^{48}$Ca.

Along this direction, we will not only improve the description of different nuclear observables consistently but also strengthen the interpretability of the supervised learning.

Acknowledgements—This work is supported by the National Natural Science Foundation of China under Grants Nos. 12005175 and 11875070, the Fundamental Research Funds for the Central Universities under Grant No. SWU119076, the JSPS Grant-in-Aid for Early-Career Scientists under Grant No. 18K13549, the JSPS Grant-in-Aid for Scientific Research (S) under Grant No. 20H05648, and the Anhui project (Z1011118169). This work is also partially supported by the RIKEN Pioneering Project: Evolution of Matter in the Universe, the RIKEN Special Postdoctoral Researchers Program, and the Science and Technology Hub Collaborative Research Program from RIKEN Cluster for Science, Technology and Innovation Hub (RCSTI).
[28] G. A. Miller, Phys. Rev. Lett. 123, 232003 (2019).
[29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Nature 323, 533 (1986).
[30] Z. Niu and H. Liang, Phys. Lett. B 778, 48 (2018).
[31] C. Ma, M. Bao, Z. M. Niu, Y. M. Zhao, and A. Arima, Phys. Rev. C 101, 045204 (2020).
[32] S. Athanassopoulos, E. Mavrommatis, K. Gernoth, and J. Clark, Nucl. Phys. A 743, 222 (2004).
[33] K. Gernoth, J. Clark, J. Prater, and H. Bohr, Phys. Lett. B 300, 1 (1993).
[34] R. Utama, W.-C. Chen, and J. Piekarewicz, J. Phys. G: Nucl. Part. Phys. 43, 114002 (2016).
[35] D. Wu, C. L. Bai, H. Sagawa, and H. Q. Zhang, Phys. Rev. C 102, 054323 (2020).
[36] X.-X. Dong, R. An, J.-X. Lu, and L.-S. Geng, Phys. Rev. C 105, 014308 (2022).
[37] G. Co’, M. Anguiano, and A. M. Lallena, Phys. Rev. C 105, 034320 (2022).
[38] R.-D. Lasseri, D. Regnier, J.-P. Ebran, and A. Penon, Phys. Rev. Lett. 124, 162502 (2020).
[39] Y. Wang, X. Zhang, Z. Niu, and Z. Li, Phys. Lett. B 830, 137154 (2022).
[40] G. A. Negoita, J. P. Vary, G. R. Luecke, P. Maris, A. M. Shirokov, et al., Phys. Rev. C 99, 054308 (2019).
[41] G. Saxena, P. K. Sharma, and P. Saxena, J. Phys. G: Nucl. Part. Phys. 48, 055103 (2021).
[42] Z. M. Niu, H. Z. Liang, B. H. Sun, W. H. Long, and Y. F. Niu, Phys. Rev. C 99, 064307 (2019).
[43] Z.-A. Wang, J. Pei, Y. Liu, and Y. Qiang, Phys. Rev. Lett. 123, 122501 (2019).
[44] C. Y. Qiao, J. C. Pei, Z. A. Wang, Y. Qiang, Y. J. Chen, N. C. Shu, and Z. G. Ge, Phys. Rev. C 103, 034621 (2021).
[45] R. Utama, J. Piekarewicz, and H. B. Prosper, Phys. Rev. C 93, 014311 (2016).
[46] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, et al. (2020) arXiv:2003.02989.
[47] J. R. Moreno, G. Carleo, and A. Georges, Phys. Rev. Lett. 125, 076402 (2020).
[48] Z.-X. Yang, X.-H. Fan, P. Yin, and W. Zuo, Phys. Lett. B 823, 136650 (2021).
[49] J. Bartel, P. Quentin, M. Brack, C. Guet, and H.-B. Håkansson, Nucl. Phys. A 386, 79 (1982).
[50] P.-G. Reinhard, in Computational Nuclear Physics 1 (Springer Berlin Heidelberg, 1991) pp. 28–50.
[51] P. Hohenberg and W. Kohn, Phys. Rev. 136, B864 (1964).
[52] K. Ryczko, D. A. Strubbe, and I. Tamblyn, Phys. Rev. A 100, 022512 (2019).
[53] J. Friedrich and P.-G. Reinhard, Phys. Rev. C 33, 335 (1986).
[54] U. C. Perera, A. V. Afanasjev, and P. Ring, Phys. Rev. C 104, 064313 (2021).
Supplemental Material for “Calibration of nuclear charge density distribution by back-propagation neural networks”

Zu-Xing Yang (杨祖星),¹ Xiao-Hua Fan (范小华),² Tomoya Naito (内藤智也),³, ⁴ Zhong-Ming Niu (牛中明),⁵ Zhi-Pan Li (李志攀),² and Haozhao Liang (梁豪兆)⁴, ³

¹ RIKEN Nishina Center, Wako 351-0198, Japan
² School of Physical Science and Technology, Southwest University, Chongqing 400715, China
³ RIKEN Interdisciplinary Theoretical and Mathematical Sciences Program, Wako 351-0198, Japan
⁴ Department of Physics, Graduate School of Science, The University of Tokyo, Tokyo 113-0033, Japan
⁵ School of Physics and Optoelectronic Engineering, Anhui University, Hefei 230601, China
I. DETAILS OF CHARGE DENSITY GENERATORS

In this section, in addition to Fig. 1, we explain the details of the three charge density generators (CDGs). The corresponding networks are built through the following 6 kinds of layer operators.

1. **Input layer $x$:**

   $$x = \{Z, N\}, \tag{S1}$$

   where $Z$ and $N$ are the proton and neutron numbers, respectively.

2. **Feature layer ($i_{12\rightarrow18}$):**

   $$\{Z, N\} \rightarrow \{Z, N, O_p, O_n, \delta, v_p, v_n, \mathcal{P}\}, \tag{S2}$$

   with

   $$O_p = Z \mod 2, \quad O_n = N \mod 2, \quad \delta = [(-1)^Z + (-1)^N]/2, \quad \mathcal{P} = v_p v_n/(v_p + v_n), \tag{S3}$$

   where $v_p$ ($v_n$) is the difference between the actual nucleon number $Z$ ($N$) and the nearest magic number, which is among 8, 20, 28, 50, 82, and 126.

3. **Fully connected layer ($i_{f,g(x)}$):**

   $$y^{(L)} = \sum_i D^{(L-1)} w^{(L)}_{ij} x^{(L)}_i + b^{(L)}_j, \tag{S4}$$

   $$x^{(L+1)}_i = g(y^{(L)}_i),$$

   where $x^{(L)}_i$ and $y^{(L)}_j$ are the inputs and outputs of the layer $L$, $D^{(L)}$ is the number of outputs, $w^{(L)}_{ij}$ and $b^{(L)}_j$ are the weights and biases, and $g(x)$ denotes the activation function, respectively. In particular, $i_{f,r}$ and $i_{f,s}$ denote $g(x) = \text{ReLU}(x) = \max(0, x)$ and $g(x) = \text{Sigmoid}(x) = 1/(1 + e^{-x})$, respectively.

4. **Convolutional layer ($i_{c,g(x)}$) [1]:**

   $$y^{(L)}_{kn} = \sum_{i,j} C^{(L)}_{in} K^{(L)}_{jn} \cdot w^{(L)}_{kl} x^{(L)}_{j,S_i n+i} + b^{(L)}_{kn}, \tag{S5}$$

   $$x^{(L+1)}_{kn} = g(y^{(L)}_{kn}),$$

   where $k \in [0, C^{(L)}_{out}]$ and $n \in [0, [D^{(L-1)}/S^{(L)}_t]]$ with $[a]$ denoting the round up of $a$, $C^{(L)}_{in}$ and $C^{(L)}_{out}$ denote the numbers of channels of input and output, respectively. Here, $C^{(L)}_{out}$ is also equal to the number of convolution kernels $w^{(L)}_{kl}$ in layer $L$. Moreover, $S^{(L)}_t$ denotes the stride when the convolution kernel slides and $K^{(L)}_s$ denotes the convolution kernel size. In this study, the one-dimensional convolution is used.

5. **Pooling layer ($i_p$):**

   $$x^{(L+1)}_n = \max(x^{(L)}_{S_i n}, x^{(L)}_{S_i n+1}, \ldots, x^{(L)}_{S_i n+K^{(L)}_s-1}), \tag{S6}$$

   where $S^{(L)}_t$ and $K^{(L)}_s$ denote the stride and size, respectively.

6. **Radius layer ($i_{p\rightarrow R}$):**

   $$R^{(L)}_{\text{pre}} = \left[ \frac{\int \rho^{(L)}_{\text{pre}}(r) r^4 \, dr}{\int \rho^{(L)}_{\text{pre}}(r) r^2 \, dr} \right]^{1/2}. \tag{S7}$$
TABLE S1. The structure and hyperparameters of CDG-1.

| L | Types and Operators | D | g(x) |
|---|---------------------|---|------|
| 0 | Input Layer (x)     | 2 | —    |
| 1 | Dense Layer (l_{f,r}) | 50 | ReLU |
| 2 | Dense Layer (l_{f,r}) | 100 | ReLU |
| 3 | Dense Layer (l_{f,r}) | 200 | ReLU |
| 4 | Dense Layer (l_{f,r}) | 400 | ReLU |
| 5 | Dense Layer (l_{f,r}) | 800 | ReLU |
| 6 | Dense Layer (l_{f,r}) | 400 | ReLU |
| 7 | Dense Layer (l_{f,r}) | 200 | ReLU |
| 8 | Output Layer (l_{f,s}) | 150 | Sigmoid |
| 9 | Radius Layer (l_{p\rightarrow R}) | 1 | — |

Other information

- Numerical normalization factor: 1000/111
- Optimizer: Adam
- Total parameters: 857,400
- Batch size: 64
- Running time: 447.543 s

TABLE S2. The structure and hyperparameters of CDG-2.

| L | Types and Operators | D | C_{in} | C_{out} | K | S | g(x) |
|---|---------------------|---|--------|--------|---|---|------|
| 0 | Input Layer (x)     | 2 | —      | —      | — | — | —    |
| 1 | Dense Layer (l_{f,r}) | 50 | — | — | — | — | ReLU |
| 2 | Dense Layer (l_{f,r}) | 100 | — | — | — | — | ReLU |
| 3 | Dense Layer (l_{f,r}) | 200 | — | — | — | — | ReLU |
| 4 | Dense Layer (l_{f,r}) | 400 | — | — | — | — | ReLU |
| 5 | Dense Layer (l_{f,r}) | 800 | — | — | — | — | ReLU |
| 6 | Convolutional Layer (l_{c,r}) | — | 5 | 4 | 2 | — | ReLU |
| 7 | Pooling Layer (l_{p}) | — | — | — | 3 | 3 | — |
| 8 | Convolutional Layer (l_{c,r}) | — | 5 | 20 | 5 | 2 | ReLU |
| 9 | Pooling Layer (l_{p}) | — | — | — | 5 | 3 | — |
| 10 | Convolutional Layer (l_{c,r}) | — | 20 | 100 | 5 | 1 | ReLU |
| 11 | Pooling Layer (l_{p}) | — | — | — | 5 | 3 | — |
| 12 | Convolutional Layer (l_{c,r}) | — | 100 | 200 | 5 | 1 | ReLU |
| 13 | Output Layer (l_{f,s}) | 150 | — | — | — | — | Sigmoid |
| 14 | Radius Layer (l_{p\rightarrow R}) | 1 | — | — | — | — | — |

Other information

- Numerical normalization factor: 1000/111
- Optimizer: Adam
- Total parameters: 567,645
- Batch size: 64
- Running time: 817.412 s

Through this layer, the density on the meshed $\rho_{\text{pre}}$ is transformed into the root-mean-square radius $R_{\text{pre}}$. The back-propagation of this layer is achieved with the help of Autograd in Pytorch [2].

The CDG-1, CDG-2, and CDG-3 are built by stacking the above unit layers and the corresponding details are listed.
TABLE S3. The structure and hyperparameters of CDG-3.

| L   | Types and Operators | D  | g(x) |
|-----|---------------------|----|------|
| 0   | Input Layer (x)     | 2  | —    |
| 1   | Feature Layer (l_{i2→i8}) | 8  | —    |
| 2   | Dense Layer (l_{f,r}) | 50 | ReLU |
| 3   | Dense Layer (l_{f,r}) | 100| ReLU |
| 4   | Dense Layer (l_{f,r}) | 200| ReLU |
| 5   | Dense Layer (l_{f,r}) | 400| ReLU |
| 6   | Dense Layer (l_{f,r}) | 800| ReLU |
| 7   | Dense Layer (l_{f,r}) | 400| ReLU |
| 8   | Dense Layer (l_{f,r}) | 200| ReLU |
| 9   | Output Layer (l_{f,s}) | 150| Sigmoid |
| 10  | Radius Layer (l_{\rho→R}) | 1  | —    |

Other information

- Numerical normalization factor: 1000/111
- Optimizer: Adam
- Total parameters: 857,700
- Batch size: 64
- Running time: 450.277 s

We extend the 2-dimensional input to 800 dimensions by linear or non-linear operations, which will spontaneously generate more features in order to extract the information of density distribution. Subsequently, this high-dimensional information is descended to produce the density distribution \( \rho_{c,i} \). During the calculations, the densities are multiplied by a numerical normalization factor, with which the densities are restricted between 0 and 1 with no effect on the root-mean-square radius. It is found that this can accelerate the speed of convergence [3]. The Adaptive Momentum Estimation (Adam) [4] is chosen as the optimizer for the stochastic gradient descent. The network prediction slightly depends on the initialization of the network parameters \( w_{\text{initial}} \), which makes it possible to derive the statistical errors by initializing them differently. All the trainable parameters are then determined, through the supervised learning of theoretical densities and the calibration based on experimental charge radii as stated in the main text. In practice, all calculations are performed on the NVIDIA GeForce RTX 3070.

**II. TRAINING AND PERFORMANCE OF CDGS**

In this section, we describe the training procedures and discuss the corresponding performance of CDG-1, CDG-2, and CDG-3 in detail.

The experimental data and the selected training nuclei are shown in Fig. S1. In addition to Fig. 3, the corresponding predicted charge radii by CDG-1 and CDG-2 are shown in Figs. S2 and S3, respectively. The corresponding difference \( \Delta R_c \) between the charge radii predicted by CDG-1, CDG-2, and CDG-3 and experiential value are shown in Fig. S4. In addition to Fig. 4, the charge radii for the Ca isotopes predicted by CDG-1, CDG-2, and CDG-3 are shown in Fig. S5. Noting that the training sets are identical for all CDGs, it can be seen that, although the overall tendency that \( R_c \) increases from \(^{39}\text{Ca}\) to \(^{44}\text{Ca}\), decreases from \(^{44}\text{Ca}\) to \(^{48}\text{Ca}\), and then increases again from \(^{48}\text{Ca}\) to \(^{50}\text{Ca}\) can be in general described by all models, the delicate odd-even staggering in \( R_c \) can only be described by CDG-3, which takes the parity, paring, and shell effects into account by the Feature Layer.

In order to further test the network predictive power, here we train CDG-3 with the training set that all the Ca isotopes are replaced by other random nuclei. The corresponding predictions are shown in Fig. S6(b) compared with Fig. S6(a) that employs the same training set as shown in Fig. S1. It is seen that, although the whole Ca isotopic chain is untrained, only a few nuclei (\(^{39}\text{Ca}, \(^{44}\text{Ca}, \text{and} \(^{48}\text{Ca}\)) cannot be covered by one standard deviation of the...
FIG. S1. (Color online) Experimental data of charge radius, where the nuclei in the training set are denoted with the blue squares.

FIG. S2. (Color online) Charge radii predicted by CDG-1.
prediction. With the same idea, Figs. S6(c) and S6(d) show the CDG-3 predictions for the Pb isotopes with and without training the Pb isotopes. It is also clear that without training the predictions for the whole Pb isotopic train (except \(^{208}\text{Pb}\)) reproduce the experimental data within one standard deviation.

The radius residual information is transferred to the correction of the density distribution by

\[
\delta \rho, \delta R = M(x, w_{\text{theo}}^{\text{MLE}} + \delta w) - (\rho, R)_{\text{theo}},
\]

\[
\delta w = \sum_{t_2} F(\frac{\partial \text{Loss}(0.7, w^{(t_2)})}{\partial w}),
\]

(S9)

where \( F \) is the Adam optimizer. The network weights \( w_{\text{theo}}^{\text{MLE}} \) to describe the theoretical distribution are biased by \( \delta w \) due to the inclusion of experimental data. The corresponding results are presented in Fig. S7. The charge density distributions of four untrained random nuclei—\(^{26}\text{Na}, ^{103}\text{Mo}, ^{152}\text{Sm}, \) and \(^{204}\text{Pb}\)—predicted by CDG-3 are shown and compared with the original SHF+BCS results, where the inserts are shown in the logarithmic scales. As can be seen from Fig. S7, the corrections are particularly small. These fine corrections are of the magnitude of \( 10^{-3} \text{ fm}^{-3} \) and converge within a very narrow interval. For different nuclei, the corrections \( \delta \rho_c \) converge to different patterns, and the correction patterns are determined by \( \delta w \). The diversity of such correction patterns may deserve further studies.

### III. DETAILS OF CALCULATING CHARGE DENSITIES IN DENSITY FUNCTIONAL THEORY

In the SHF+BCS theory, we calculate the matter densities using the equation

\[
\rho_\tau(r) = \sum_i w_i |\phi_i(r, \sigma, \tau)|^2,
\]

where \( w_i \) denote the occupancy probability for each single-particle state and \( \tau = p \) (proton) corresponds to proton (neutron) [5]. With the spherical symmetry, the single-particle wave function reads

\[
\phi_i(r) = R_i(r) Y_{jlm}(\theta, \psi),
\]

where \( Y_{jlm}(\theta, \psi) \) is a spinor spherical harmonics. Taking into account that the nucleon itself has an intrinsic electromagnetic structure, we need to fold the proton and neutron densities with the intrinsic charge density of nucleons. Such a folding becomes a product in Fourier space, so we transform the densities into the so-called form factors

\[
F_\tau(k) = 4\pi \int_0^\infty dr r^2 j_0(kr) \rho_\tau(r),
\]

(S10)
FIG. S4. (Color online) The difference between the charge radii predicted by CDGs and experiential value.
where $j_0$ is the spherical Bessel function of the zeroth order. Similarly, the form factor of the spin-orbit current $\nabla \cdot J$, accounting for the magnetic contributions to the charge density, is written as $F_{ls,\tau}(k)$. The charge form factor is then given by

$$F_c(k) = \sum_{\tau} [F_{\tau}(k)G_{E,\tau}(k) + F_{ls,\tau}(k)G_{M,\tau}(k)] \exp \left[ \frac{(hk)^2}{8 \langle P^2_{cm} \rangle} \right],$$

(S11)
where $G_{E,\tau}(k)$ and $G_{M,\tau}(k)$ are the electric and magnetic form factors of nucleons, respectively, and the exponential factor is the center-of-mass correction. See Ref. [5] for details. Finally, the charge density is obtained from the charge form factor by the inverse Fourier-Bessel transform,

$$\rho_{c,\text{theo}}(r) = \frac{1}{2\pi^2} \int_0^\infty dk k^2 j_0(kr) F_c(k).$$  \hspace{1cm} (S12)

**IV. DETAILS OF CDTBE MAP**

In this section, in addition to Fig. 5, we explain the details of the feedforward neural network for the map from charge density to binding energy (CDTBE).

There are three parts in the CDTBE map—M1, M2, and M3. M1 is the previously trained charge density generator, and we take the CDG-3; M2 is a map from charge density to matter density; M3 is a map from the neutron, proton, and charge densities to the binding energy per nucleon.

On the one hand, M2 is composed of 5 cells ($\hat{C}_{1,2,3,4,5}$), which can be regarded as the inverse of Eqs. (S10), (S11), and (S12). We denote the model as

$$\begin{bmatrix} \hat{C}_4 \\ \hat{C}_5 \end{bmatrix} \begin{bmatrix} \hat{C}_3(\hat{C}_1(x) \uplus \hat{C}_2(\rho_c)) \end{bmatrix} = \begin{bmatrix} \rho_p \\ \rho_n \end{bmatrix}.$$  \hspace{1cm} (S13)

Here, the symbol $\uplus$ indicates splicing two vectors, e.g., $[1,3,2] \uplus [2,5,7] = [1,3,2,2,5,7]$. The detailed structure of M2 is listed in Table S4.

On the other hand, M3 is composed of 4 cells ($\hat{C}_{1,2,3,4}$). We denote the model as

$$\hat{C}_4 \left[ \hat{C}_3(\hat{C}_1(x) \uplus \hat{C}_2(\rho_{p,n,c})) \right] = E/A.$$  \hspace{1cm} (S14)

The detailed structure of M3 is listed in Table S5.

In the present study, M2 and M3 are trained by the densities and binding energies calculated by SHF+BCS with different effective interactions, SkM, SkM*, SkIII, SLy4, SkT, and SkT3. As a result, the obtained learning deviations
| L  | Types and Operators | D  | g(x) | |
|----|---------------------|----|------|---|
| 0  | Input Layer (x)     | 2  |      |   |
| 1  | Dense Layer (l_{f,r}) | 32 | ReLU | |
| 2  | Dense Layer (l_{f,r}) | 64 | ReLU | |
| 3  | Dense Layer (l_{f,r}) | 128| ReLU | |

| L  | Types and Operators | D  | C_{in} | C_{out} | K_s | S_t | g(x) | |
|----|---------------------|----|--------|--------|-----|-----|------|---|
| 0  | Input Layer (\rho_c) | 150|        |        |     |     | ReLU | |
| 1  | Convolutional Layer (l_{c,r}) | 1  | 8     | 3     | 3   | ReLU |      | |
| 2  | Pooling Layer (l_p)     |     |       | 2     | 2   |     | ReLU | |
| 3  | Convolutional Layer (l_{c,r}) | 8  | 32    | 5     | 5   | ReLU |      | |
| 4  | Dense Layer (l_{f,r})   | 512|       |       |     |     | ReLU | |

| L  | Types and Operators | D  | g(x) | |
|----|---------------------|----|------|---|
| 0  | Dense Layer (l_{f,r}) | 512| ReLU | |
| 1  | Dense Layer (l_{f,r}) | 256| ReLU | |
| 2  | Dense Layer (l_{f,s}) | 150| Sigmoid | |

| L  | Types and Operators | D  | g(x) | |
|----|---------------------|----|------|---|
| 0  | Dense Layer (l_{f,r}) | 512| ReLU | |
| 1  | Dense Layer (l_{f,r}) | 256| ReLU | |
| 2  | Dense Layer (l_{f,s}) | 150| Sigmoid | |

Other information

|                                |       |
|--------------------------------|-------|
| Numerical normalization factor | 6.25  |
| Optimizer                      | Adam  |
| Total parameters               | 2,763,404 |
| Batch size                     | 32    |
| Running time                   | 2241.901 s |

are less than 0.2%.
### TABLE S5. The structure and hyperparameters of M3.

| L | Types and Operators | D | $g(x)$ |
|---|---------------------|---|--------|
| 0 | Input Layer ($x$) | 2 |        |
| 1 | Dense Layer ($l_{f,r}$) | 32 | ReLU |
| 2 | Dense Layer ($l_{f,r}$) | 64 | ReLU |
| 3 | Dense Layer ($l_{f,r}$) | 128 | ReLU |

**Cell-2 ($C_2 = \{I_{c,r}, l_p, l_{c,r}, l_{f,r}\}$)**

| L | Types and Operators | $C_{in}$ | $C_{out}$ | $K_s$ | $S_t$ | $g(x)$ |
|---|---------------------|--------|--------|-------|-------|--------|
| 0 | Input Layer ($p_{p,n,c}$) | 150 | 3 | — | — | — |
| 1 | Convolutional Layer ($l_{c,r}$) | — | 3 | 24 | 3 | 3 | ReLU |
| 2 | Pooling Layer ($l_p$) | — | — | — | 2 | 2 | — |
| 3 | Convolutional Layer ($l_{c,r}$) | — | 24 | 72 | 5 | 5 | ReLU |
| 4 | Dense Layer ($l_{f,r}$) | 512 | — | — | — | ReLU |

| L | Types and Operators | $g(x)$ |
|---|---------------------|--------|
| 0 | Dense Layer ($l_{f,r}$) | 768 | ReLU |
| 1 | Dense Layer ($l_{f,r}$) | 1024 | ReLU |

**Cell-3 ($C_3 = \{l_{f,r}\}$)**

| L | Types and Operators | $g(x)$ |
|---|---------------------|--------|
| 0 | Dense Layer ($l_{f,r}$) | 1024 | ReLU |
| 1 | Dense Layer ($l_{f,r}$) | 512 | ReLU |
| 2 | Dense Layer ($l_{f,r}$) | 256 | ReLU |
| 3 | Dense Layer ($l_{f,r}$) | 128 | ReLU |
| 4 | Dense Layer ($l_{f,r}$) | 32 | ReLU |
| 5 | Dense Layer ($l_{f,s}$) | 1 | Sigmoid |

**Other information**

| | |
|---|---|
| Numerical normalization factor | 6.25 |
| Optimizer | Adam |
| Total parameters | 2,177,241 |
| Batch size | 32 |
| Running time | 2740.545 s |
[1] K. Simonyan and A. Zisserman, (2014), arXiv:1409.1556.
[2] N. Ketkar, in Deep Learning with Python (Apress, 2017) pp. 195–208.
[3] Z.-X. Yang, X.-H. Fan, P. Yin, and W. Zuo, Phys. Lett. B 823, 136650 (2021).
[4] D. P. Kingma and J. Ba, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, edited by Y. Bengio and Y. LeCun (2015).
[5] P.-G. Reinhard, in Computational Nuclear Physics 1 (Springer Berlin Heidelberg, 1991) pp. 28–50.