Machine-Learning-Assisted Quantum Control in a Random Environment

Tangyou Huang,[1,2] Yue Ban,[2,3] E. Ya. Sherman,[2,4] and Xi Chen[2,*]

[1] *International Center of Quantum Artificial Intelligence for Science and Technology (QuArtist) and Department of Physics, Shanghai University, Shanghai 200444, China*
[2] *Department of Physical Chemistry, University of the Basque Country UPV/EHU, Bilbao, Spain*
[3] *School of Materials Science and Engineering, Shanghai University, Shanghai 200444, China*
[4] *IKERBASQUE Basque Foundation for Science, Bilbao 48013, Spain*

(Received 1 September 2021; revised 4 January 2022; accepted 21 January 2022; published 14 February 2022)

Disorder in condensed matter and atomic physics is responsible for a great variety of fascinating quantum phenomena, which are still challenging for understanding, not to mention the relevant dynamical control. Here we introduce *proof of the concept* and analyze a neural-network-based machine-learning algorithm for achieving feasible high-fidelity quantum control of a particle in random environment. To explicitly demonstrate its capabilities, we show that convolutional neural networks are able to solve this problem as they can recognize the disorder and, by supervised learning, further produce the policy for the efficient low-energy cost control of a quantum particle in a time-dependent random potential. We show that the accuracy of the proposed algorithm is enhanced by a higher-dimensional mapping of the disorder pattern and using two neural networks, each properly trained for the given task. The designed method, being computationally more efficient than the gradient-descent optimization, can be applicable to identify and control various noisy quantum systems on a heuristic basis.

DOI: 10.1103/PhysRevApplied.17.024040

I. INTRODUCTION

Machine learning (ML), which enables computers to learn automatically from available task-specific data [1–4], is revolutionizing modern approaches in physical sciences [5]. In quantum science, ML becomes useful and powerful [6] in particle physics, many-body physics [7], and quantum computing [8] among others. Recently developed learning architectures [9] such as convolution neural networks (CNNs), having a considerable success in object detection and image classification, were beneficial to classify phases of matter [10], study nonequilibrium glasses [11], find hidden order in electronic-quantum-matter imaging data [12], and identify the thermodynamic time arrow [13].

All of the above studies were performed for systems where disorder is either nonexisting or plays a negligible role in the system dynamics. In practice, impurities, noise, and other imperfections are ubiquitous and unavoidable in condensed matter [14] and its simulated counterparts [15]. Particularly, the ultracold atoms offer a feasible and controllable platform for studying the disorder [16–19]. In this scenario, the random potential is implemented by optical means, and brings about a variety of intriguing phenomena [20–24], i.e., localization effects, phase transitions, and superfluidity, due to the interplay among the disorder, nonlinearity, trapping potential, or/and spin-orbit coupling. Along with these developments, the power of supervised learning (SL) is harnessed to categorize stochastic data, extract quantitative information from this data, and predict the features of complex quantum systems, at a reasonable computational cost [25–30].

However, quantum control under disorder still remains a major challenge [31–35], though optimal control [36–38], ML [25,39–42], and shortcuts to adiabaticity [43,44] have been exploited for fast manipulations in regular systems. The extensive study of stochastic systems [45,46] have emerged in the quest for controlling the dissipative dynamics most efficiently. However, when it comes to disorder, to classify or identify stochastic data embodied in the dynamics is a conundrum. As the size of the stochastic sample increases dramatically, the higher power of ML is demanding in such complexity.

To work out this problem, we establish the ML approach for identifying and controlling dynamics of a quantum system with disorder. For this purpose, we use deep learning with two CNNs for high-fidelity control of a quantum particle in a time-varying trapping potential embedded in random environment. We begin with a useful result: training the CNN can efficiently preselect the relevant type of the disorder realization from tens of thousands of stochastic samples. Then, we introduce the second CNN to find the

*chenxi1979cn@gmail.com

optimal control policy such as the time-dependent potential shape, in a training regression model. To make the optimization more efficient, the randomness classification from deep learning is an essential pretraining for disordered system under control, thus removing the redundant data. Thus, the SL with CNNs provides the ability to generalize the tasks beyond their original design, applicable to any realization of random potential. Our methods pave an efficient way for the robust optimal control, i.e., cooling, transporting, trapping atoms, or charged particles (ions and electrons) [39,43,47], by taking into account environmental noise and randomness.

II. DISORDERED SYSTEM AND CONTROL STRATEGY

Consider a quantum particle of mass $m \equiv 1$, located at the sum of time-dependent harmonic potential and a random potential of impurities. The corresponding Hamiltonian (with $\hbar \equiv 1$) reads

$$H(t) = \frac{p^2}{2} + \frac{1}{2}\omega^2(t)x^2 + U_r(x), \tag{1}$$

where p is the momentum, $\omega(t)$ is the frequency of harmonic trap, and $U_r(x)$ is the random potential of interest. Equation (1) describes atoms in optical traps and electrons in acoustic traps [47] and gate-formed quantum dots. The motivation behind the frequency modulation, i.e., from $\omega(t \le 0) = \omega_0$ to $\omega(t = t_f) = \omega_f$, is to achieve the fast high-fidelity expansion and compression within a short time t_f, beyond the adiabatic criteria [39,43].

We study generic random potential $U_r(x)$, corresponding to the Anderson-like disorder, produced by $\mathcal{N} \gg 1$ impurities at the positions $x_j = x_{j-1} + d$ regularly separated by the distance d. The potential can be presented in the following form:

$$U_r(x) = U_0 \sum_{j=1}^{\mathcal{N}} s_j u(x - x_j), \tag{2}$$

with $u(z) = \exp(-z^2/\xi^2)$. Here U_0 is the amplitude potential of a single impurity, and $s_j = \pm 1$ is a random function of j with mean values $\langle s_j \rangle = \langle U_r(x) \rangle = 0$, and correlators $\langle s_j s_l \rangle = \delta_{jl}$, $\langle U_r(x)U_r(x') \rangle = \sqrt{\pi/2}U_0^2\xi \exp\left[-(x-x')^2/2\xi^2\right]/d$. Each disorder realization is a random sequence of ± 1, e.g., $S_i[j] = \{1, -1, 1 \cdots 1\}$, with i and j being the realization number and impurities position, respectively.

We consider narrow impurities, where the width ξ satisfies condition $\xi \ll U_0^{-1/2}$ and the corresponding localization length at the impurity with $s_j = -1$ is of the order of $1/(U_0\xi) \gg d$. Thus, localization by disorder involves many impurities [32] while the interaction energy with a single impurity behaves as approximately $U_0\xi s_j|\psi(x_j)|^2$, where $\psi(x)$ is the wave function. For a sufficiently strong parabolic potential $\omega^2x^2/2$, the ground state has the energy close to $\omega/2$ and the harmonic oscillator (ho) width $w_{\rm ho} \sim 1/\sqrt{\omega}$. As the potential fluctuations behave as $\sqrt{N_{\rm imp}}$, where $N_{\rm imp} \sim 1/(d\sqrt{\omega})$ is the number of impurities at the localization length of the state, we estimate the shift in the ground-state energy as $\Delta\epsilon/\omega \sim U_0\xi/(\sqrt{d}\omega^{3/4})$. To estimate the length ℓ of the disorder-induced localization, we minimize the sum of the kinetic energy approximately $1/\ell^2$ and potential energy in the disorder potential as $\sim U_0\xi/\sqrt{\ell d}$ and obtain $\ell \sim (U_0\xi/\sqrt{d})^{-2/3}$ with the corresponding energy $\epsilon_{\rm loc} \sim (U_0\xi/\sqrt{d})^{4/3}$. Therefore, in the parabolic potential, localized states can be located at the distances up to $w_d \sim \sqrt{\epsilon_{\rm loc}}/\omega \sim (U_0\xi/\sqrt{d})^{2/3}/\omega$, meaning that with the decrease in ω, the ground state can be positioned at a large distance from the origin.

Figure 1 illustrates that the eigenstates of the final trap can be completely changed by different realizations of random potential, as compared to the disorder-free results. For the realization in Fig. 1(b), where the initial and final states are almost orthogonal, the high-fidelity results cannot be achieved even with the optimal control policy presented below. This intriguing feature makes the previous methods [39,43] invalid in our current problem. As a consequence, we need improved statistical analysis and computational method.

To proof the principle of ML application we choose the third-order polynomial

$$\omega(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3, \tag{3}$$

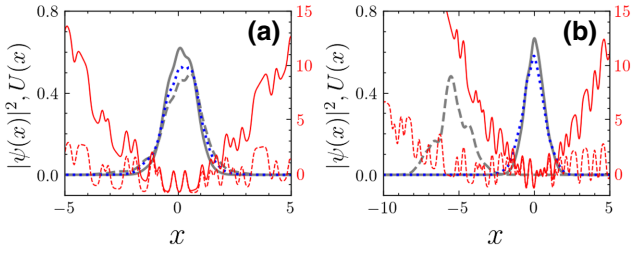

FIG. 1. Probability densities of the initial ($t = 0$, black solid line) and final ($t = t_f$, black dashed line) ground states in the harmonic trap in the random environment forming the total potential $U(x)$. Two realizations are presented to illustrate the effect of disorder. The corresponding final states (blue dotted lines) produced by the optimal control policy with SL are shown as well. The total initial (red solid line) and final (red dashed line) potentials, are also shown for the eye. Parameters: $U_0 = 1$, $\omega_0 = 1$, and $\omega_f = 0.1$. Here and below we use $\xi = d = 1/8$ for $\mathcal{N} = 160$ impurities at the $\{-10, 10\}$ interval. Since we are using the system of units with $\hbar \equiv m \equiv 1$, the length and the energy are measured in units of $1/\sqrt{\omega_0}$ and ω_0, respectively.

as the control function for the trap frequency, where $a_0 = \omega_0$, and $a_3 = \left[\omega_f - (\omega_0 + a_1 t_f + a_2 t_f^2)\right]/t_f^3$ are given by the boundary conditions, $\omega(0) = \omega_0$ and $\omega(t_f) = \omega_f$. The initial state at $t = 0$ is assumed to be the ground state for simplicity. The freedom left in a_1 and a_2 offers the possibility to optimize the control function $\omega(t)$, thus finding the maximum ground-state fidelity defined as

$$F \equiv \left| \int_{-\infty}^{\infty} \psi^*(x, t_f)\psi_{\rm gr}(x|\omega_f)dx \right|^2, \tag{4}$$

where $\psi_{\rm gr}(x|\omega_f)$ is the ground state in the random potential corresponding to ω_f, and $\psi(x, t_f)$ is obtained by a direct numerical solution of the nonstationary Schrödinger equation with the Hamiltonian $H(t)$ from Eq. (1). The optimal design of the trap frequency through the control policy $A = \{a_1, a_2\}$ can produce $\psi(x, t_f)$ with the maximum possible fidelity.

We impose two conditions on the optimal control function, with the hint from the analysis on the high-fidelity control without disorder in Appendix A. First, it has to provide a high fidelity for the quantities of interest, in this case, as defined in Eq. (4). Second, $\omega^2(t)$ should correspond to a moderate energy consumption required for the transition, suggesting that the maximum $\omega^2_{\rm max}(t)$ does not exceed a certain value Ω^2 such that the process is experimentally feasible. Figure 2 illustrates the high-fidelity zone control policy $A = \{a_1, a_2\}$ and corresponding feasible control function $\omega(t)$, satisfying the criteria $\omega^2_{\rm max}(t) \leq \Omega^2 = 6$. The search for the optimal coefficients in the relevant $\{a_1, a_2\}$ range (see Fig. 2) is a time-consuming task

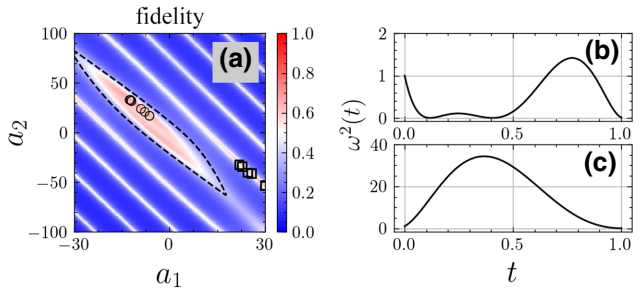

FIG. 2. (a) The fidelity of the control policy $A = \{a_1, a_2\}$ for disorder-free harmonic potential (the blue-pink background) with the high-fidelity zone (dashed line) satisfying the criteria $\omega^2_{\rm max}(t) \leq \Omega^2$. The "feasible" and "unfeasible" (with $F \geq F_b = 0.9$, where F_b is the fidelity bound) control policies in the presence of disorder are indicated by "○" and "□" symbols. Two example functions of $\omega^2(t)$ are compared in (b) and (c), corresponding to the "feasible" and "unfeasible" solutions. Parameters: $\omega_0 = 1$, $\omega_f = 0.1$, $t_f = 1$, and $\Omega = \sqrt{6}$, taken here as an example. Interestingly, for a given set $A = \{a_1, a_2\}$ the fidelity in the presence of disorder can be higher than that for the disorder-free harmonic potential. The parameters a_1 and a_2 are measured in the units of ω_0^2 and ω_0^3, respectively.

even for a given realization. Since the stationary state and dynamics rely on the disorder realizations, the optimization of control policy also requires immense computing power. Note that the total number of disorder realizations in Eq. (2) is approximately $2^{\mathcal{N}}$. However, in agreement with the manifold hypothesis [48], many of these realizations produce similar $U_r(x)$ functions with similar $\psi_{\rm gr}(x|\omega_f)$ width and positions. Therefore, the ML can use databases of moderate ($\leq 10^5$) size. In what follows, we are motivated to develop the SL based on two CNNs to overcome such a challenge.

III. MACHINE-LEARNING PROCEDURE

Now, we proceed to use SL, comprising two CNNs, for classifying the disorder realizations and constructing the optimal control policy, through the connection between the random sequence $S_i[j]$ and the optimal control policy $A_{\rm opt}$, see the schematic diagram in Fig. 3. One can refer to Appendix B 1 for the technical description of SL.

First, we generate 4×10^4 disorder realizations with the labeled sequences $S_i[j]$ as the inputs. For each realization, the fidelity of overlap between the eigenstates at $t = t_f$ and final wave functions resulting from the state evolution [see Eq. (4)] is numerically calculated with the control function $\omega(t)$ in Eq. (3). The maximum fidelity for the given ith realization, $F_i^{\rm max}$, and corresponding control policy A_i are thus determined by using the same approach in Fig. 2, where the criteria $\Omega^2 = 6$ and $F_b = 0.9$ are applied to bound the feasibility and fidelity, while keeping a considerable size of database. The whole database $X = \{S_i, F_i^{\rm max}, A_i\}$ is finally established, where 80% of the database is selected as a training set, and the rest as a testing set.

Then, we introduce the first CNN, named in what follows CNN1, in deep learning to assign each given realization of the random potential to a set of classes, for instance,

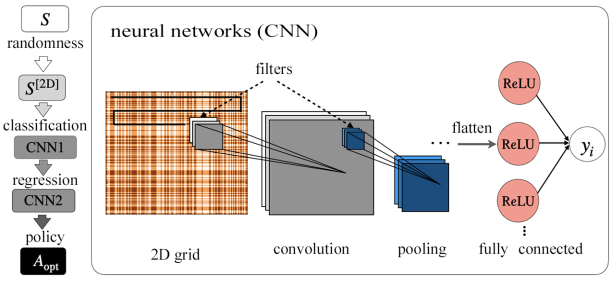

FIG. 3. Schematic diagram (left) of SL with two CNNs for randomness recognition and regression. Working flow (right) of CNN includes conversion from a 1D $S_i[j]$ to a 2D grid $S_i^{[2D]}[j_1, j_2]$, convolution and pooling layers, fully connected layer with the ReLU activation function and the output y_i. Details, including description of the ReLU function, are presented in Appendix B.

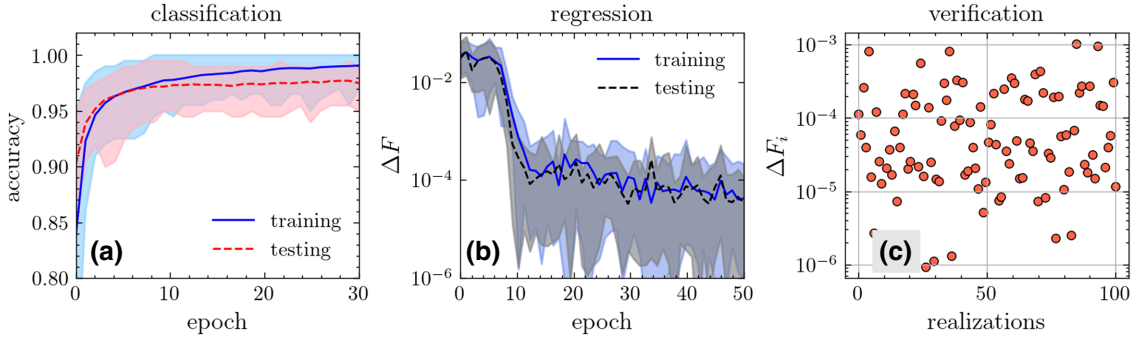

FIG. 4. The accuracy of CNN1 (a) and the fidelity deviation (b) are displayed for classification and regression, where the dashed and solid lines represent the average value of test and training batches in each epoch. The shadow area indicates the value distribution of batches. (c) The fidelity deviation from two trained CNNs are presented for 100 testing realizations of random potential.

whether it determines feasible high fidelity (FH) or not. Such randomness recognition is classification, aiming at selecting the reasonable inputs of realizations. To be more efficient, we extend the input $S_i[j]$ into a two-dimensional (2D) grid (see Appendix B 2) before the neural network is trained, by converting each sequence $S_i[j]$ into a 2D matrix $S_i^{[2D]}[j_1, j_2]$ by using

$$S_i^{[2D]}[j_1, j_2] \equiv S_i[j_1] + S_i[j_2]. \tag{5}$$

As expected, the randomness recognition based on the 2D grid surpasses the one-dimensional (1D) one, in the sense that the accuracy of classification and loss of regression are improved at the cost of computation time. Therefore, we use $S_i^{[2D]}[j_1, j_2]$ as the inputs and y_i as the output, where FH ($y_i = 1$) and anti-FH ($y_i = 0$) suggests the aforementioned criteria, $F > F_b = 0.9$ and $\omega_{\max}^2(t) \leq 6$, is satisfied or not.

For classification in the CNN1 we employ the standard sequential structure (convolution and pooling layers), and choose the loss function as $L_1(y, p) = -\sum_i [y_i \log(p_i)]$, with p_i being the probability produced by network, and the accuracy N_r/N, with N_r being the number of the right predictions out of total N. After using optimizer `Adam()` at the rate of 10^{-4}, we manage to select 5886 out of 4×10^4 realizations, with the accuracy above 97%, see Fig. 4(a) and the relative portion of the selected realizations being of the order of the $w_{\text{ho}}/w_d \sim \omega^{1/2}/(U_0\xi/\sqrt{d})^{2/3}$ ratio. Obviously, this pretraining process is critical for classifying the disorder and excluding realizations yielding the low-fidelity control, as shown in Fig. 1(b). Remarkably, the high efficiency of CNN1 can be conceptually interpreted by comparing its feature map with the corresponding position of final wave packet, also see the detailed discussion in Appendix C 2.

Next, to find the optimal control policy A_{opt}, we construct the second CNN (CNN2) for regression. We choose the loss function $L_2(y, y') = \sum_i (y_i - y_i')^2/N$, where y and y' are the actual and predicted results of control policy A. The residual neural network [49] is used in CNN2, with a shortcut channel. We define the fidelity deviation for each realizations $\Delta F_i = |F_i^{\max} - F_i'|$ with F_i' here being the fidelity predicted by the CNN-based control policy. Further, we define the average value over each N-sized batch $\Delta F = \sum_i^N \Delta F_i/N$ in every training epoch for quantifying the performance of CNN2. As a consequence, we train the CNN2 for achieving $\Delta F \leq 10^{-4}$, see Fig. 4(b). Thereby, during the process we record the loss at each batch and the fidelity of predicted policies, and finally obtain the trained CNN1 and CNN2, as indicated by solid lines in Figs. 4(a) and 4(b). Moreover, we produce 100 realizations for verifying the performance of trained CNNs in Fig. 4(c), and also discuss the dependence of their efficiency on the hyperparameter in Appendix B 3. Accordingly, after training two CNNs with 4×10^4 input disorder realizations, the optimal control policy A_{opt} to design $\omega(t)$ for the high-fidelity control with any random potential is obtained.

IV. DISCUSSION

There are several points to be addressed on the generality of our proposed method. We can, in principle, choose other $\omega(t)$ ansatzes with more parameters or even use the results from the gradient-descent optimization. The detailed analysis clarifies that the influence of the form of ansatz (or moderate changing in the bound F_b and/or Ω) on the classification of disorder, performed by the CNN1, is essentially negligible, since the border line between high and low fidelity is mostly determined by the intrinsic property such as the shape of the disorder rather than by the external condition. However, malfunctioning or poor performance of CNN1 can cause low efficiency of CNN2, obtaining the input from CNN1. The ansatz (3) serves as a reference for setting the criteria. Note that the CNNs trained with the gradient-descent optimization is not better than the ones with such a simple ansatz, see the detailed discussion in Appendix C 1.

Moreover, we can also apply the trained CNNs to different values of t_f and ω_f. Figure 5 indicates the average

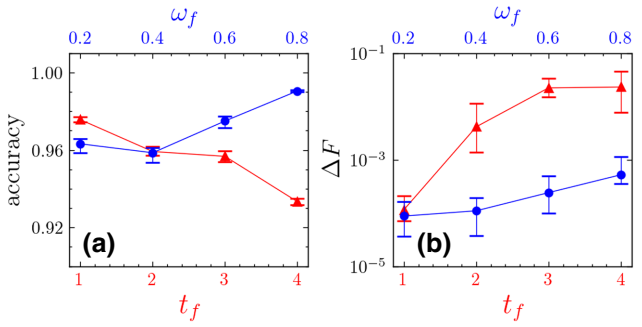

FIG. 5. The average accuracy in CNN1 (a) and the average fidelity deviation in CNN2 (b) for the last ten epochs are illustrated for different ω_f and t_f, where the structure and hyperparameters are the same as those in Fig. 4, and the error bars represent their deviations.

accuracy in CNN1 and the average fidelity deviation in CNN2 for the last ten epochs by using the same structure and hyperparameter as before. On the one hand, when ω_f is increased, the random realizations are much easier to recognize, thus resulting in higher accuracy. It makes sense that the influence of random potentials on the fidelity can be negligible, when the trap potential is strong enough to localize the state near the origin. However, the more realizations as the inputs of CNN2 finally lead to the larger fidelity deviation as shown in Fig. 5. On the other hand, according to the time-energy trade-off, larger t_f (still far away from the adiabaticity) increase the area corresponding to condition $\omega^2_{\max}(t) \leq \Omega^2$ (cf. Fig. 2). Thus, more random realizations corresponding to the feasible A_{opt} increase the statistical uncertainty and degrade the performance of trained CNNs. That is, the fidelity deviation in CNN2 becomes larger because of worse classification, depending on the distribution and number of the selected realizations in CNN1, see Fig. 5. In other words, the combined effects of the trapping potential and disorder plays a role in dynamical control, characterized by the fidelity and the required energy cost, e.g., the laser power for optical trap or the electrical power for quantum dots.

V. CONCLUSIONS

The behavior of quantum objects such as atoms and charged particles in random potentials is an active research area, with a lot of the accumulated knowledge and even more yet unknowns. The complexity prevents the researchers from efficiently controlling the quantum dynamics in random environments. We presented a remedy by developing *proof-of-principle* supervised learning algorithms, trained through deep neural networks, to classify the randomness and find the optimal control policy. The efficiency and accuracy of the proposed algorithm is based on using two-dimensional mapping of the random potential and sequential application of two neural networks, each trained for the given different task. Our results indicate that machine learning, based on the convolutional neural network for classification and regression, can be used to control various quantum systems with impurities, noise, and imperfections, and ultimately to unveil the physical insight into the interplay of disorder and quantum dynamics. With the advent of techniques of configurable optical traps [50] and surface acoustic waves [51,52], we suggest the experimental verification of the proposed method for trapped atoms or electrons in random environment.

ACKNOWLEDGMENTS

This work is supported from National Natural Science Foundation of China (12075145), Science and Technology Commission of Shanghai Municipality (2019SHZDZX01-ZX042019 and 20DZ2290900), SMAMR (2021-40), Program for Eastern Scholar, CSC fellowship (202006890071), Basque Government IT986-16, PGC2018-095113-B-I00, PGC2018-101355-B-I00 funded by MCIN/AEI/ 10.13039/501100011033 and by "ERDF A way of making Europe", EU FET Open Grant Quromorphic (828826) and EPIQUS (899368), and the Ramon y Cajal program (RYC-2017-22482).

APPENDIX A: HIGH-FIDELITY QUANTUM CONTROL WITHOUT DISORDER

To begin, we consider the case without random potential, $U_r(x) \equiv 0$, in order to have a reference for understanding the effects induced by the disorder. By setting $m \equiv 1$ and $\hbar \equiv 1$, the Hamiltonian of a single particle trapped in a harmonic potential reads

$$H = \frac{p^2}{2} + \frac{1}{2}\omega^2(t)x^2, \tag{A1}$$

which describes the compression and decompression by tailoring the frequency $\omega(t)$ of the harmonic trap. According to the Lewis-Riesenfeld invariant theory, the solution of time-dependent Schrödinger equation admits analytical expression [43]:

$$\psi(x,t) = \left(\frac{\omega_0}{\pi b^2}\right)^{1/4} \exp\left[-\frac{i}{2}\int_0^t \frac{\omega_0}{b^2}dt'\right] \times \exp\left[i\frac{1}{2}\left(\frac{\dot{b}}{b} + i\frac{\omega_0}{b^2}\right)x^2\right], \tag{A2}$$

where the auxiliary function $b(t)$ satisfies the Ermakov equation:

$$\ddot{b} + \omega^2(t)b = \frac{\omega_0^2}{b^3}. \tag{A3}$$

For a decompression process from the initial frequency ω_0 to final frequency ω_f, the boundary conditions can

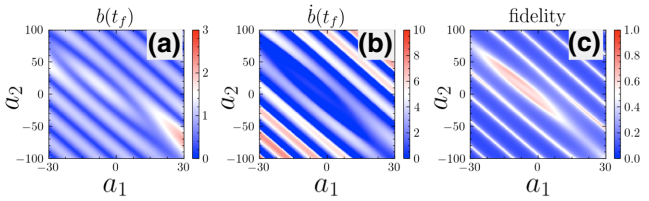

FIG. 6. Dependence of $b(t_f)$, $\dot{b}(t_f)$, and the fidelity F on the coefficient grid $\{a_1, a_2\}$, where the parameters are $\omega_0 = 0$, $\omega_f = 0.1$, and $t_f = 1$. The straight lines corresponding to the maximum fidelity can be obtained with an approximate solution of Ermakov Eq. (A3) as $a_2 = -3a_1/t_f + 12n\pi/t_f^3$, with integer n.

be formulated as $b(0) = 1, b(t_f) = \gamma$ ($\gamma = \sqrt{\omega_0/\omega_f} > 1$), $\dot{b}(0) = \dot{b}(t_f) = 0$. Thus, for an arbitrary control function $\omega(t)$, we are able to calculate the time-dependent scaling parameter of $b(t)$ and corresponding $\dot{b}(t)$ by solving the Ermakov equation. By considering the ground state, with the initial and final boundary conditions, we, in general, can reach the ideal target state, that is, $\psi_{\rm gr}(x|\omega_f) = (\omega_0/\pi\gamma^2)^{1/4} e^{-\omega_0 x^2/(2\gamma^2)}$. Based on Eq. (4), the fidelity thus can be analytically expressed as

$$F = \left[\frac{4\omega_0^2 b_f^2 \gamma^2}{\omega_0^2(\gamma^2 + b_f^2)^2 + (\dot{b}_f b_f \gamma^2)^2}\right]^{1/2}, \tag{A4}$$

where $b_f = b(t_f)$ and $\dot{b}_f = \dot{b}(t_f)$ are the numerical solution of Eq. (A3) at $t = t_f$. Obviously, the fidelity F strongly depends on b_f and $\dot{b}_f$. When $b_f = \gamma$ and $\dot{b}_f = 0$, we will have $F = 1$. In this case, we recall the concept of shortcuts to adiabaticity, that is, to achieve fast adiabaticlike decompression without final excitation.

Without loss of the generality, we choose the simple ansatz $\omega(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$, such that the fidelity is calculated, depending on the coefficients a_1 and a_2. To understand the performance of the fidelity, Figs. 6(a) and (b) illustrate the dependence of $b(t_f)$ and $\dot{b}(t_f)$ on $\{a_1, a_2\}$, respectively. Figure 6(c) finally shows the plot of the fidelity dependent of the coefficient set $A = \{a_1, a_2\}$, in which the stripes occur due to the interplay between $b(t_f)$ and $\dot{b}(t_f)$. The high-fidelity regime in Fig. 6 gives the criteria for machine learning later, when the random potential is involved.

APPENDIX B: MACHINE LEARNING

Machine-learning (ML) methods, including support vector machines, decision trees, random forests, and artificial neural networks (ANNs), have been developed in recent decades. Moreover, the deep learning is proposed to handle the huge quantity of data and complex system, notably, the ANNs usually outperform the others. Nowadays, the ANNs are dedicated to solving complex tasks such as the image and video recognition, analysis of strategical games (AlphaGo), etc. In particular, the deep CNNs, initially proposed for computer vision learning, now are overwhelming in the artificial-intelligence (AI) industry. Their unique architecture, inspired by research on the brain's visual cortex, greatly enhances the performance of analysis of systems in complex surroundings, which is consistent with our problem on quantum control in a random environment.

The reasons for using the CNNs to analyze the disordered system are threefold: (1) Data grows exponentially with a tremendous amount of disorder realizations; (2) training an ANN can be accelerated by using graphic processor units (GPUs); (3) CNN can be used to identify the disorder as the application in image classification.

Next, we exploit the supervised learning, based on two CNNs, for classifying and controlling the joint effect of a regular (parabolic) potential and disorder.

1. Neural network and supervised learning

A deep ANN consists of input, hidden, and output layers, and the depth of network usually depends on the amount of hidden layer. Meanwhile, a single layer is composed by a set of nodes, and each node is connected with the others from the next layer with a particular weight and bias. Moreover, the learning process of ANNs is combined by the forward-propagation and back-propagation computation based on the gradient-descent algorithm. We start with the propagating data from the input layer, pass the hidden layer(s), measure the output layer, and finally calculate network error based upon the network predictions. With the error function and the gradient-base optimizer, the backpropagation decreases error by updating the weights and bias of network. Compared with a regular ANNs, the CNNs are trained to optimize the filters (or kernels) through the automated learning, instead of the hand engineered in feature extraction. It takes advantage of the hierarchical pattern in capturing data feature and reducing the number of the parameters involved. In order to explain the functioning of this CNN, we make use of the following notation:

1. x^ℓ is the data flow of ℓth layer.
2. The filter K with the size $k_1 \times k_2$ has m and n as the iterators.
3. The weight between ℓ layer and $\ell - 1$ layer is represented by ω^ℓ, and the corresponding bias b^ℓ.
4. $f(\cdot)$ is an activation function.
5. The underlying data of layer is $x_{i,j}^\ell = \sum_{m,n} f(w_{m,n}^\ell x_{m,n}^{\ell-1} + b^\ell)$, where i and j are the iterator.
6. $x^\ell \otimes K^k$ represents the data extracting process by the $k^{\rm th}$ filters.
7. y_i and y_i' are the actual and predicted values (labels), respectively.

Supposing that we use k filters, the output of ℓth convolutional layer can be presented as

$$x_{i,j}^{\ell} = \sum_{k=0}^{k} x_{i,j}^{\ell-1} \otimes K^{k} = \sum_{k=0}^{k} \sum_{m,n}^{k_1,k_2} f\left(K_{m,n}^{k} x_{i+m,j+n}^{\ell-1} + b^{\ell}\right), \tag{B1}$$

where the activation function $f(\cdot)$ is the logistic `Sigmoid` function, $f(z) = 1/[1+\exp(-z)]$, or the rectified linear unit (`ReLU`) function, $f(z) = \max(0, z)$. The `Sigmoid` function maps the data from $[-\infty, +\infty]$ into $[0, 1]$, resulting in the probability of prediction as the output of network. And the `ReLU` is a piecewise step function, `ReLu(x)` $= \max(x, 0)$, that transfers the input data from $[-\infty, +\infty]$ into $[0, +\infty]$. Two such nonlinear activation functions are widely used to allow the nodes to learn more complex structures in the data. A pooling layer, aiming to reduce the spatial size, contains MaxPooling() and AveragePooling(). More specifically, they extract the maximum (or average) value of the pooling block from the previous layer, thus reducing the amount of the parameters. The CNN layer is schematically shown in Fig. 7, in which we set the 16×16 inputting data and three 7×7 filters for the convolution layer and three 2×2 filters for calculating the maximal pooling.

Next, we introduce the loss function and gradient-based optimizer for classification and regression. Regarding the classification task, the loss function is defined as the following cross-entropy form:

$$\mathcal{J}(W, b; y, y') = \frac{1}{N} \sum_{i=1}^{N} J_1(W, b; y_i, y_i'), \tag{B2}$$

with

$$J_1(W, b; y_i, y_i') = -y_i \log\left[\sigma(y_i')\right], \tag{B3}$$

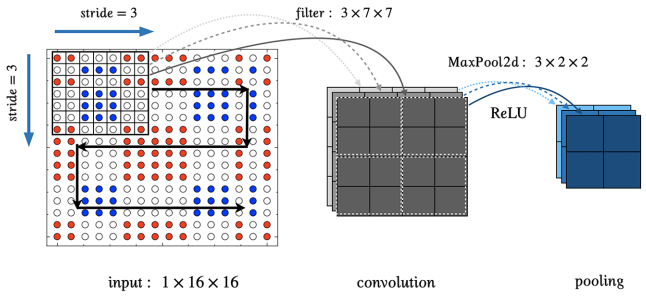

FIG. 7. A single unit of CNN includes the convolution, activation, and pooling process. We take 16×16 grids as an example for illustrating the working flow and variables in the function Conv2d() and Maxpool(). In this case, the process can be represented by Conv2d(1,3,7,3) and MaxPool2d(2) in the PyTorch.

where W is the weight collection of network for N samples, and $\sigma(y_i')$ is the softmax probability, where the Softmax function $\sigma(z_i) = e^{z_i}/(\sum_j e^{z_j})$ is used for normalizing the output. As for the two-category image classification $k = 2$ task, the input layer is a flatten pixel sequence x_i of image, and the result is the probability of labels. For instance, when the actual binary label is $y_0 = \{1, 0\}$, and two-dimension output $y_0' = \{p_0, p_1\}$, the error for a single prediction thus is $j = -y_0 \log[y_0']^T$. On the other hand, for the regression process, the loss function in Eq. (B2) is a mean-squared error:

$$J_2(W, b; y_i, y_i') = \sum_i |y_i - y_i'|^2, \tag{B4}$$

which represents the deviation from the regression prediction y_i' to the actual sample y_i. We use the optimizer Adam(), which is included in the application programming interface (API) of PyTorch, for optimizing the loss function in the learning process. Backpropagation (or forward pass) refers to the calculation and storage of the intermediate variables (weights and bias) of a neural network, and minimizes the cost function by a gradient-based optimizer. This can be simply expressed as

$$\texttt{Repeat}: \left\{ W_{i,j}^{\ell} = W_{i,j}^{\ell-1} - \eta \frac{\partial \mathcal{J}}{\partial W_{m,n}^{\ell}} \right\}, \tag{B5}$$

with learning rate η.

Following that, we create the algorithm for our task, which consists of two CNNs for classification and regression, respectively. We encode the algorithm based on the PyTorch [53] software platform, where the deep-learning library consists of the tensor flow and the computation is accelerated by GPUs. In order to illustrate the learning algorithm, we briefly introduce the functions that we used in the PyTorch API:

1. 2D convolution layer: `Conv2d(inchannel, outchannel, kernel.size, stride)`.
2. Max pooling layer `MaxPool2d(kernelsize)`.
3. `ReLU` and `Sigmoid` represent the rectified linear unit function and the corresponding logistic function, respectively.
4. `CrossEntropyLoss()` and `MSELoss()` indicates the loss function of Eqs. (B2) and (B4).

The variables include the following: *inchannel*, the depth of channel in the input; *outchannel*, the number of output channel depends on the amount of filter (or kernel); *kernelsize*, the filter size; *stride*, controlling the stride for the cross-correlation.

The detailed parameters can be further found in the PyTorch tutorial [53]. Along with this user-friendly platform, we now construct the algorithm for the supervised

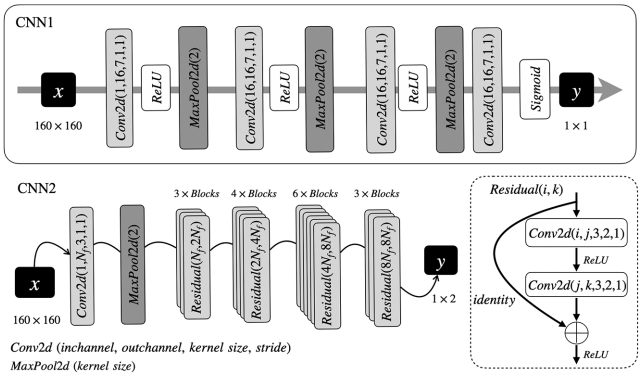

FIG. 8. Diagrammatic architectures of CNN1 and CNN2, are illustrated, where the function and its parameters are presented for each layer of network and the residual block of CNN2 in the dashed frame is specified. More details can be found in the main text.

learning. Before proceeding, we should design the architecture of CNN1 and CNN2, since the performance of a neural network mostly depends on its structure and layer depth. According to the complexity of task, the architecture of CNN1 is built up as a standard sequential network and CNN2 as a *ResNet* network [49], see the details in the flow chart in Fig. 8. The residual block `Residual()`, with so-called "identity shortcut connection," skips two layers, as shown in Fig. 8. It makes the network possible to train hundreds of layers, keeping the compelling performance. After introducing the CNN-based supervised learning and the architecture of two networks, we can start with creating the database and training the model for classification and regression.

2. Classification and regression

For supervised learning, two essential steps, including data preparation and model training, are required. In this sense, the performance of model can be improved by increasing the training data and selecting a high-quality database. To calculate the database, however, is a time-consuming task for a complex system, so it is significant to preselect for producing a representative database with high quality. Let us consult Fig. 1 of the main text, in which the eigenstates of the final trap can be completely changed by different realizations of random potential, and some of them will result in the low-fidelity control for sure. Thus, we propose CNN1 for the preselection, in order to establish the link between input and output data of network by choosing a small amount of high-quality database. We demonstrate that the high-quality database not only brings the benefits to training process, but also makes the trained network more universal and tolerant.

Aiming to present the feature of each single random sequence, we initially extend the one-dimension sequence in the two-dimension grid, see Eq. (5). More specifically, as in the main text, we select a 1×160 random sequence, e.g., $S_i[j] = \{1, 1, -1, 1, \ldots, -1, 1\}$. A typical resulting 2D grid with the elements 2,0, and -2, is shown in Fig. 9(d). We see the advantage of 2D $S_i^{[2D]}[j_1, j_2]$ as the input data, in the following discussion.

Next, we generate 4×10^4 realizations of disorder, and thus calculate the maximum fidelity $F_i^{\max}$ and the corresponding policy A_i of the 200×200 coefficient grid in the range of $a_1 \in [-30, 30]$ and $a_2 \in [-100, 100]$. Here we set two conditions for the optimal control function. First, it has to provide a high fidelity for the quantities of interest, i.e., $F_{\max} > F_b$. Second, the corresponding $\omega^2(t)$ should correspond to a moderate energy consumption required for the transition, implying that $\omega^2_{\max}(t)$ has not exceeded a certain value Ω^2. In practice, by taking into account the experimental constrains, such as the limited laser intensity or the gate field in quantum dots, we set the control policy, $\omega^2_{\max}(t) \leq \Omega^2$, where we take $\Omega = \sqrt{6}$ as the critical value for defining "feasible" policy. In Fig. 9(a), the contour curve for $\omega^2_{\max}(t) = 6$ is presented. Moreover, the optimal policy A_{opt} is constrained by these two conditions: $F_{\max} > F_b$ and $\omega^2_{\max}(t) \leq \Omega^2$ (labeled by FH). The ratio of the database as a function of F_b is also presented in Fig. 9(b), from which we find that the amount of FH database is decreased when we set larger bound, F_b, for the fidelity. Obviously, the disorder effect makes

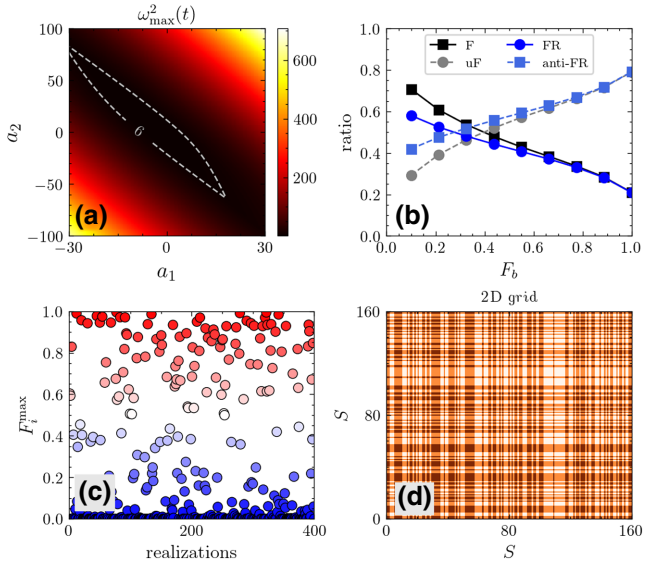

FIG. 9. (a) Dependence of the maximum value of $\omega^2(t)$ on the coefficient set $\{a_1, a_2\}$, and the white dashed curve presents the contour of $\omega^2(t) \equiv 6$. (b) The proportion of four classifications, based on two criteria [$F > F_b$ and $\omega^2_{\max}(t) \leq \Omega^2$], in the prepared database as the function of F_b. The distribution of maximum fidelity $F_{\max}$ for 400 exemplified realizations is plotted in (c), and one of realization in the 2D grid is illustrated in (d). Here the criteria $F_b = 0.9$ and $\Omega = \sqrt{6}$ are used, and other parameters as the same as those in Fig. 6.

the fidelity worse, though the higher fidelity is desirable in the quantum control in the presence of random environment. In order to keep balance between the amount of high-fidelity realizations and the diversity of database, we set the bound $F_b = 0.9$ as the criteria for keeping the reasonable database, see Fig. 9(c). With the assistance of the prepared database satisfying such criteria, we discuss the network and training process as follows.

Previously, we attempted to find the regression between $S_i^{[2D]}[j_1, j_2]$ and $A_{\rm opt}$ by using only one CNN. However, the results are not reasonable, and a very complex neural network is required to provide the expressibility and universality for the variety of disorder realization. Nevertheless, we create an intuitive scheme to reduce the complexity of database, that is, the classification is added prior to the regression. The database is divided into two categories by the pretraining process: the realization satisfying feasible high-fidelity (labeled FH) criteria or not (labeled anti-FH). As a consequence, the database for regression is firstly filtered by the classification (CNN1) process based on two aforementioned criteria, and secondly train the network (CNN2) based on previously identified FH database.

Now, we train the CNN1 with the input $X = S_i^{[2D]}[j_1, j_2]$ and output $Y = \{0, 1\}$ by selecting the loss function `CrossEntropyLoss()` with respect to Eq. (B2). The identified FH database from CNN1 is the input data of CNN2, and the output is optimal policy $A_{\rm opt}$ with the loss function `MSELoss()` in Eq. (B4). Two architectures of CNNs are presented in Fig. 8, where there are seven layers in a regular sequential network CNN1 and 34-layer ResNet34 [49] for CNN2. Meanwhile, we use the optimizer `Adam()` [54] to optimize the parameters based on the gradient descent algorithm. Moreover, we define the `Accuracy` $= N_r/N$ (with N_r being the number of the right predictions out of total N) for CNN1, which is the correct prediction number over the total amount of database. Meanwhile, for quantifying the result of regression, we also define the fidelity deviation $\Delta F = |F_i^{\rm max} - F_i'|$, where $F_i^{\rm max}$ is the actual maximum fidelity and F_i' is the numerical result from the policy predicted by the network (as in Sec. III of the main text).

To this end, we formulate the training algorithm as follows:

Input: The database $\{x, y\}$
Output: Trained CNN
initialization;
optimizer=Adam(learning rate = 0.0001);
loss = CrossEntropyLoss()(or MSELoss());
while epoch **do**
  for batch in range(epoch size) **do**
    net.train(),
    predications = net(x_i[*batch*]),
    training loss = loss(predications, y[*batch*]),
    optimizer(net),
    total loss += training loss,
Average Loss = total loss/epoch size,
epoch+ = 1,
end
end

Algorithm 1. Training CNN for classification and regression

3. Machine-learning outcome

In this section, we present a detailed training process and further discuss the results. To proceed with the training and testing, we choose the parameters, such as $\omega_0 = 1$, $\omega_f = 0.1$, and $t_f = 1$. The coefficients in the control function of $\omega(t)$ are in the range of $a_1 \in [-30, 30]$ and $a_2 \in [-100, 100]$, and the classification criteria are $F_b = 0.9$ and $\Omega = \sqrt{6}$. The whole database $X = \{S_i, F_i^{\rm max}, A_i\}$ for 4×10^4 realizations in the 200×200 coefficient grid are established by a 50-core computer for more than 10 h. The input data for CNN1 is a 2D random grid $x_i = \{S_i[2D]\}$ and the output is $y_i = \{0, 1\}$, to classify the optimal policy is FH ($y = 1$) or anti-FH ($y = 0$). Remarkably, CNN1 manages to select 5886 realizations out of 4×10^4, when the criteria, $F > F_b = 0.9$ and $\omega_{\rm max}^2(t) \le \Omega^2 = 6$, are stipulated. Eventually, we convert these classified realizations into the CNN2 as the input database, and the corresponding optimal policy $A_{\rm opt}$ is obtained as the output data. For both of the two networks, 80% of the input database is the training database and the rest testing part. One can find other parameters in Fig. 8 and more details in the code.

It turns out that the accuracy of CNN1 can reach 97% after 30 iterations (epoch = 30), and the fidelity deviation for CNN2 is below 10^{-4} after 50 iterations (epoch = 50). The average loss of training and testing data are presented by the solid and dashed curves in Figs. 10(a) and 10(b), where we see that the overfitting occurs at 10 epoch for classification, and at 20 epoch for regression.

After that, we discuss the generality of our method and the tolerance of model for changing the hyperparameters. First of all, we compare the performance of two trained CNNs by using 1D and 2D input data. The accuracy and fidelity deviation ΔF for 1D and 2D input data are presented in Figs. 10(c) and 10(d), where `Conv1d()` and `Conv2d()` are exploited for 1D and 2D cases, and the rest parameters are the same. It is evident that the model using 2D input data outperforms the 1D model in terms of accuracy and fidelity deviation.

Second, we elaborate the generality of our training model by checking the performance with various values of ω_f and t_f. To this end, we prepare the databases of 1.6×10^4 realizations for $t_f = \{1, 2, 3, 4\}$ and $\omega_f = \{0.2, 0.4, 0.6, 0.8\}$, the criteria and parameters of two CNNs are the same as the previous case when $t_f = 1, \omega_f = 0.1$. In Fig. 11, we specify the maximum fidelity located in the whole database for the various conditions, where (a) $\omega_f =$

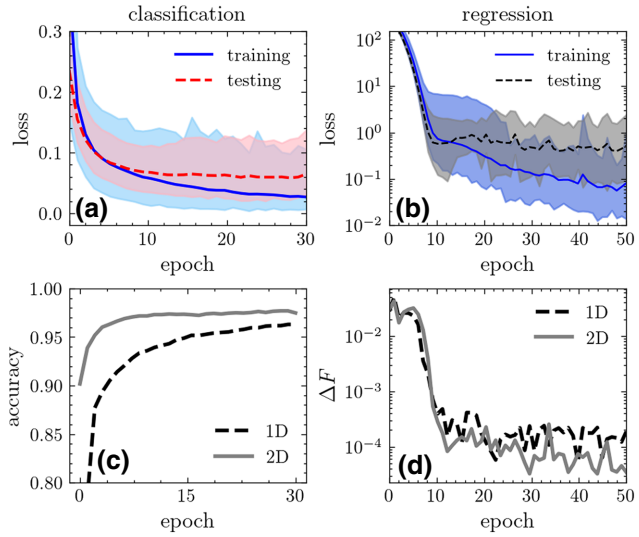

FIG. 10. Training and testing loss as a function of epochs in CNN1 (a) and CNN2 (b) for classification and regression. The performances of CNN1 (c) and CNN2 (d) are compared by using 1D (dashed curve) and 2D (solid curve) input data. Here the shadows are the values of the batches in each epoch.

$0.1, t_f = 1$, (b) $\omega_f = 0.4, t_f = 1$, and (c) $\omega_f = 0.1, t_f = 2$ are considered. By comparison, the larger ω_f results in the higher fidelity, since the random realizations are much easier to recognize, when the final trap frequency is increased. This is due to the fact that the influence of random potentials on the fidelity can be negligible, when the final trap potential is strong enough such that the localized state has to be located near the origin. Consequently, the lower loss of CNN1 is achieved since most of disorder realizations are labeled as FH, in contrast, more inputs cause the performance of CNN2 to degrade. In addition, according to the time-energy trade-off, the increase of total time t_f makes the designed trap frequency easier to satisfy the predetermined criteria [$F > F_b = 0.9$ and $\omega^2_{\max}(t) \leq \Omega^2 = 6$], yielding the larger area in Fig. 7(c). In this case, the database is difficult to recognize, see Fig. 11, since more random realizations corresponding to the feasible A_{opt} increase the statistical uncertainty and degrade the performance of trained CNNs. Therefore, the loss of CNN1 becomes larger when the total time t_f, but the loss of CNN2 decreases conversely. All these results are consistent with those of accuracy and fidelity deviation in Fig. 5 of the main text. Clearly, the quantity and quality of the database determine the performance of CNNs, depending on the physical constraints or conditions, or the total time, the amplitude of disorder, and trapping potential. We conclude that the interplay of the trapping potential and disorder is of critical significance for controlling the dynamics in terms of the fidelity and the required energy.

Finally, we check the performance of the deep CNNs in terms of the hyperparameter, such as the number of

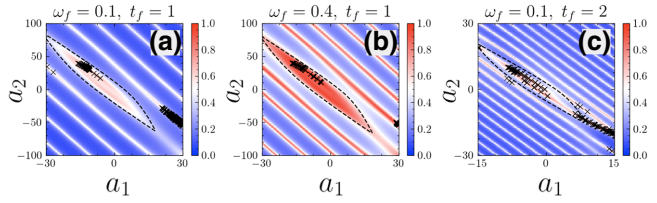

FIG. 11. Fidelity as a function of the coefficient grid $\{a_1, a_2\}$ for various ω_f and t_f, where (a) $\omega_f = 0.1, t_f = 1$, (b) $\omega_f = 0.4, t_f = 1$, and (c) $\omega_f = 0.1, t_f = 2$ are considered. The location of the maximum fidelity (black cross) is specified for 3.2×10^4 realizations of disorder in each plot. The restriction imposed by $\Omega = \sqrt{6}$ is illustrated by black dashed curve in (a), (b), and (c). The other parameters are the same as those in Fig. 6.

hidden layers, the size and number of filters, etc. In our model, the depth of the CNN2 is much larger than that of CNN1, which suggests that the CNN2 is more sensitive to the hyperparameters. For simplicity, we concentrate on two hyperparameters, the filter number and hidden layers, in the CNN2. In this network, the first layer's outchannel number is N_f (see Fig. 8), which determines the total number of filters. With different $N_f = [4, 8, 16, 32]$, we compare the average loss of testing data for 10-layer ResNet10, 18-layer ResNet18, 34-layer ResNet34, and 50-layer ResNet50. The clear dependence on these hyperparameters is presented in Figs. 12(a) and 12(b), in which the corresponding average training and testing losses of the last 10 epochs are calculated by using same parameters, respectively. Obviously, the expressibility of network depends on the number of parameters. The average training loss decreases when the number of layers or filters increases. However, we emphasize that the over fitting of the network appears when the network complexity (the number of nodes and alternative paths) increases, see Fig. 12(b). Here we note that all calculations are implemented by using the online computation resource from Google's cloud service called "*Colab*," which contains GPU's acceleration. For 30 epochs, it takes about 300 s

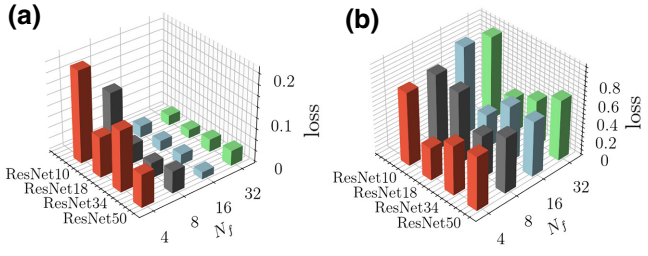

FIG. 12. Average loss of training (a) and testing (b) data for different layer number and N_f in CNN2. The average loss is the average one of last 10 epochs among 50 epochs in the training process. The other parameters are the same as those in Fig. 8. Noting we here use another database with the same size of 4×10^4 for clarifying the effect of hyperparameters.

for training the CNN1, but more than 10^3 s for the CNN2 while calculation of the fidelity deviation ΔF takes several hours. The suggested algorithm can be realized at a regular computer without GPU's acceleration albeit with a much longer computation time.

APPENDIX C: DISCUSSIONS

1. Gradient-descent optimization

Here we discuss the generality of the ansatz used here in our proposed method. One might be interested to try other ansatz and even an optimal (or near-optimal) approach, combined with ML. Regarding the latter, a powerful numerical tool, for example, the gradient-descent (GD) algorithm can be applied directly, not as a working tool of the ML algorithms. To clarify the advantages and disadvantages of this approach, let us study the possible trade-off on the improvement of fidelity in the problem of interest and the ability of training CNNs. Thus, we compare the optimal solutions produced by GD with the polynomial ansatz-based results.

A parametric optimization problem is the minimization of a given cost function by gradient descent. The optimal solution $M^{\rm opt}$ can be produced by minimizing cost value $c = J(M)$, which can be expressed as

$$M^{\rm opt} = \min_{c} J(M). \tag{C1}$$

In our scenario, the control function is the trap frequency, $f(t) = \omega(t)$, with N_t intervals' discrete time $t \in [0, t_f]$ (keeping the same $t_f = 1$ as that in the main text). Accordingly, the control tuple $f(t) = \{f(0), f(dt), \ldots, f(t_f)\}$ is constrained by $|f(t)| \leq \Omega = \sqrt{6}$ and satisfies boundary conditions, e.g., $f(0) = 1$ and $f(t_f) = 0.1$. Then, we optimize the N_t-size tuple $f(t)$ for approaching the highest fidelity by minimizing the infidelity $1 - F$, in the context of parametric constrained minimization problem. In this regard, we perform the optimization process by algorithm `SLSQP` [55] based on the `scipy` platform. For one typical realization of random potential, the GD takes several minutes to search the optimal control function, which satisfies the convergent condition ($|dJ/dM| < 10^{-7}$) of the cost function while our two-step supervised learning method produces the near-optimal solution in several seconds. Next, we are concerned about the efficiency of training two CNNs by using GD-produced databases.

To this end, we calculate the GD-based control function for the same 4×10^4-realization database used in the main text. It is expected that the GD method with $N_t = 100$ improves the fidelity. Thus, it increases the number of FH realizations, thus providing 6801 of them against 5886 for the ansatz-based method. In Fig. 13 we present the fidelity distribution of 500 realizations in (a) for two methods: GD (red circle) and ansatz

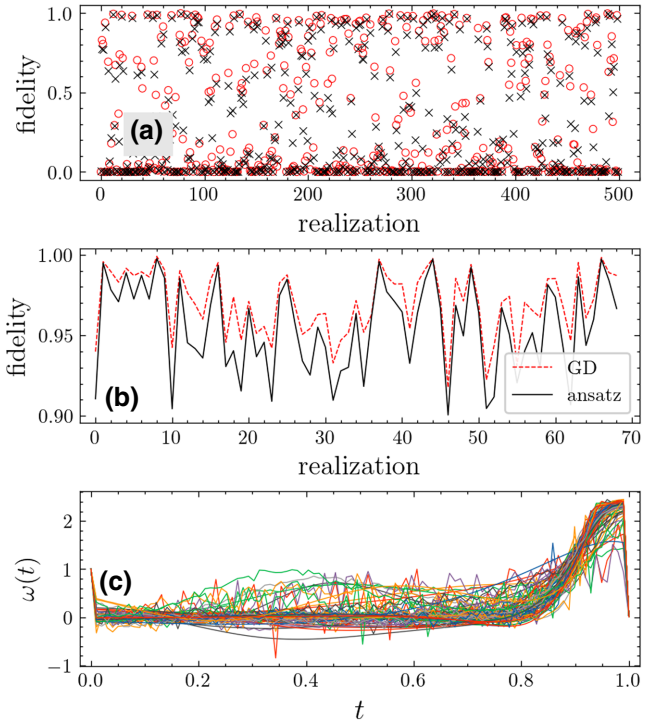

FIG. 13. Fidelity distribution for 500 realizations produced by a GD-based (red circle) and ansatz-based (black cross) scheme in (a), and 69 high-fidelity realizations satisfying high-fidelity ($F > 0.9$) by both methods are illustrated in (b). We present corresponding GD-based control functions in (c). Parameters: $N_t = 100$, others that the two methods share are the same as those in Fig. 6.

based (black cross). More distinctly, we compare 69 realizations among 500, which admit the high fidelity for both methods in (b) of Fig. 13, with the corresponding 69 optimal solutions produced by GD illustrated in (c). One can see that, although the GD-based method slightly increases the fidelity compared to the ansatz-based one, it does not change the fidelity distribution strongly. This result can be understood by the physics argument that the fidelity of control policy depends mainly on the localization induced by random potential rather than on the control strategy. Figure 14 further demonstrates the performance of CNN1 (classification) and CNN2 (regression) trained by the two databases generated from a simple ansatz (blue solid) and GD (black dashed). In addition, one can see the disadvantages of GD-based optimal control as the database for training CNN2. The GD method indeed boosts the fidelity of control policy on the cost of loosing the generality in CNN2. It is due to the fact that the performance of GD-based CNN2 is worse: the database dimension $N_t = 100$ is much larger than that for the ansatz-based database (which is 2), eventually decreasing the reliability of the regression process. Thus, the balance between the fidelity

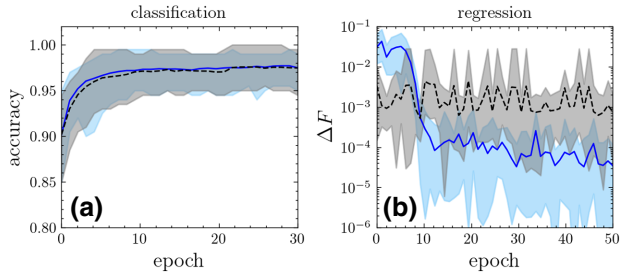

FIG. 14. Databases generated for training two CNNs by two techniques: GD (black dashed) and ansatz based (blue solid). Left: the accuracy of classification for testing data versus training epoch. Right: ΔF of testing data versus training epoch for regression. In both subfigures, the corresponding shadowed area contains the result of training batches in each epoch, and curves are the average values. The shared parameters are the same as those in the main text.

improvement and the ability to train the CNN should be kept as our method.

2. Interpretability of CNNs

It is difficult to explain the results obtained from ML in an intuitive way, despite many successful applications in quantum physics [1,10]. In order to understand the machine-making decision in solving the optimal control problem, we discuss the interpretability (or explainability) of a ML task. The interpretability in ML is defined, for example, by Miller [56]: *"Interpretability is the degree to which a human can understand the cause of a decision"* or, similarly, by Kim [57] as *"Interpretability is the degree to which a human can constantly predict the model's result."* The interpretability of a training model brings criteria such as comprehensibility, reliability, and fairness of facts upon the process of ML. In a recent work [58], Molnar offers a comprehensive review on the concept, principles, and importance of explainable models in the field of ML. Among them, we offer here the evidence of interpretability by visualizing the feature map of CNN1 for understanding and explaining the ML outcomes [58].

First, we recall the element of output from the convolution operation `Conv2d()`:

$$x_{i,j}^{\ell} = \sum_{m,n}^{k_1,k_2} K_{m,n}^{k} x_{i+m,j+n}^{\ell-1} + b^{\ell}, \tag{C2}$$

where the ℓth feature map $x_{i,j}^{\ell}$ is the sum of the product of filters $K_{m,n}^{k}$, and corresponding filter-size $(l-1)$th feature map $x_{i+m,j+n}^{\ell-1}$ with bias b^{ℓ}. According to the structure of CNN1 designed in Fig. 8, we have 16 7×7 weight matrices (filters) in each convolution layer. In Fig. 15, we present 16 parametric filters of the last layer in (a) and corresponding bias in (e), and produce 16 feature maps for three selected realizations in (b)–(d) after `Sigmoid` function. For illustration, we extract the most representative feature maps (labeled by black dashed squares) out of 16 in (e)–(h) of Fig. 15, and compare them with the corresponding density of the final wave packet with the trap frequency $\omega(t_f) = \omega_f$. By performing the four-layer convolution product operation, an original input 2D random grid (see Fig. 9) is transformed into a particular feature map, which can be interpreted by the localization of the target state density. More specifically, the feature map is strongly correlated with the localization of density for low-fidelity realization, such as (e) and (g) in Fig. 15. For the high-fidelity case, the feature map is much more uniformly distributed compared to the low-fidelity counterparts. In Fig. 16, we further compare the final-state probability

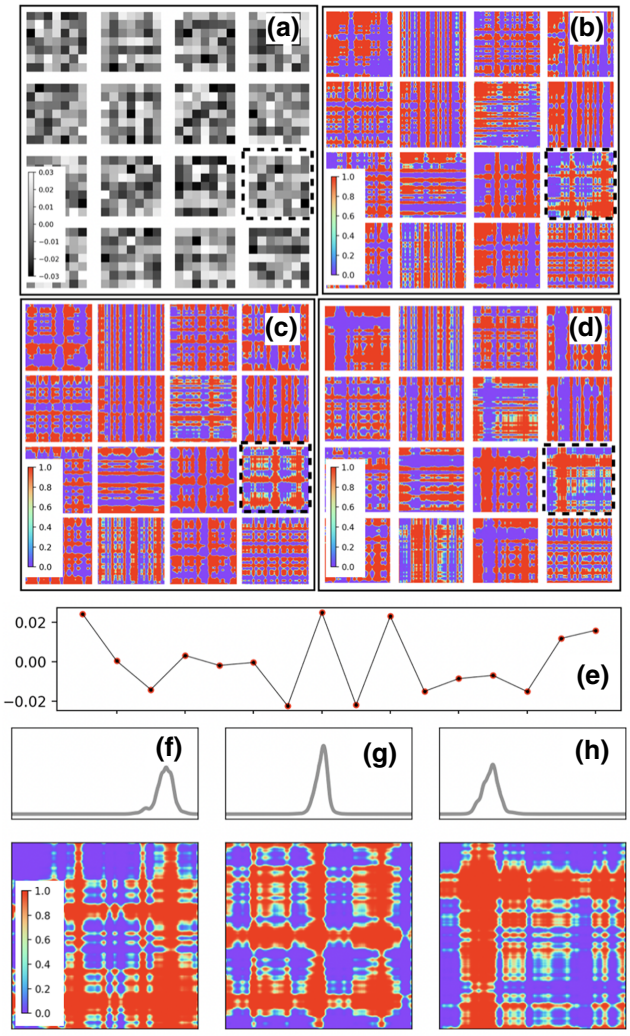

FIG. 15. The 16 parametric filters of the fourth layer for CNN1 in (a) and related bias in (e). (b)–(d) The feature map for three different realizations, and corresponding densities of wave packets and the selected feature map (labeled by black dashed squares) in (f)–(h), respectively.

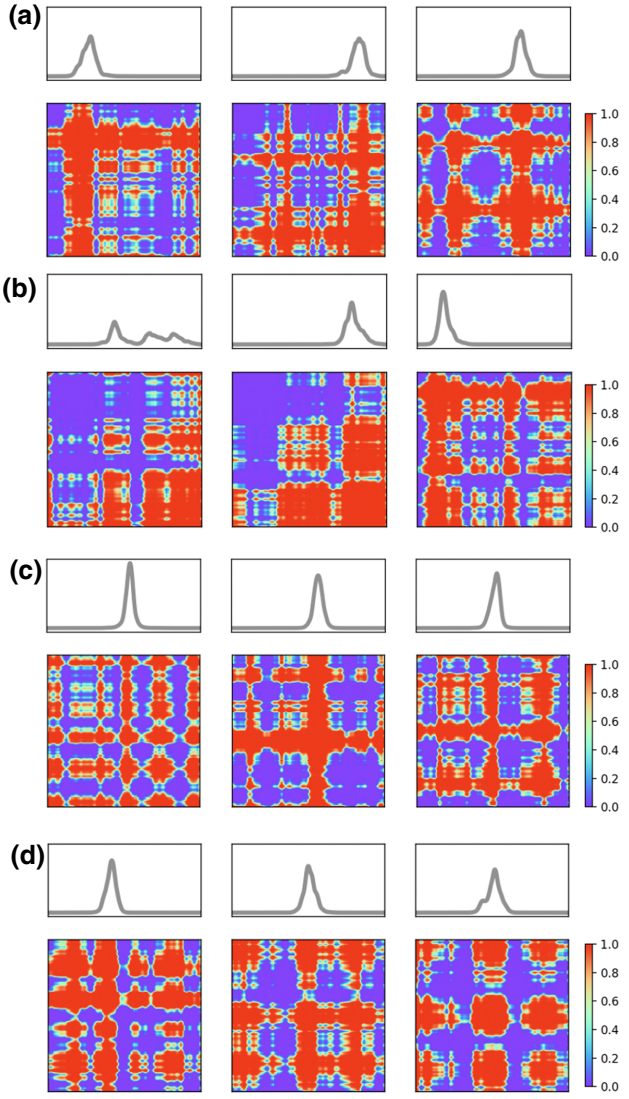

FIG. 16. Wave-packet density and related feature map selected as in Fig. 15 for 12 realizations. Panel (a) and (b) for low-fidelity ($F < 0.9$) and (c) and (d) for high-fidelity ($F > 0.9$) cases.

density and feature map for 12 realizations including low-fidelity (a) and (b) and high-fidelity (b) and (d) realizations. To this end, one cannot precisely identify the random sequence just by watching the feature map, in particular, for realizations with F_b close to neither 1 nor 0. However, for realizations with fidelity $F \ll 1$ or $F \to 1$ can be easily identified and explained, according to the typical feature map in (e)–(g) of Fig. 15. It should be emphasized that our results can be interpreted based on the comparison of the feature map and the wave-packet density. In other words, the accurate ML outcome captures the hints from the feature maps, which are related to the nature of the localization physics in random potentials.

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. **25**, 1097 (2012).

[2] S. Lawrence, C. L. Giles, Ah Chung Tsoi, and A. D. Back, Face recognition: A convolutional neural-network approach, IEEE Trans. Neural Netw. **8**, 98 (1997).

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, and Georg Ostrovski, *et al.*, Human-level control through deep reinforcement learning, Nature **518**, 529 (2015).

[4] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, and Marc Lanctot, *et al.*, Mastering the game of go with deep neural networks and tree search, Nature **529**, 484 (2016).

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning* (MIT Press, Boston, MA, 2016).

[6] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová, Machine learning and the physical sciences, Rev. Mod. Phys. **91**, 045002 (2019).

[7] Giuseppe Carleo and Matthias Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[8] Thomas Fösel, Petru Tighineanu, Talitha Weiss, and Florian Marquardt, Reinforcement Learning with Neural Networks for Quantum Feedback, Phys. Rev. X **8**, 031084 (2018).

[9] Jürgen Schmidhuber, Deep learning in neural networks: An overview, Neural. Netw. **61**, 117 (2015).

[10] Juan Carrasquilla and Roger G. Melko, Machine learning phases of matter, Nat. Phys. **13**, 431 (2017).

[11] Samuel S. Schoenholz, Ekin D. Cubuk, Efthimios Kaxiras, and Andrea J. Liu, Relationship between local structure and relaxation in out-of-equilibrium glassy systems, Proc. National Acad. Sci. **114**, 263 (2017).

[12] Yi Zhang, A. Mesaros, K. Fujita, S. D. Edkins, M. H. Hamidian, K. Ch'ng, H. Eisaki, S. Uchida, J. C. Séamus Davis, Ehsan Khatami, and Eun-Ah Kim, Machine learning in electronic-quantum-matter imaging experiments, Nature **570**, 484 (2019).

[13] Alireza Seif, Mohammad Hafezi, and Christopher Jarzynski, Machine learning the thermodynamic arrow of time, Nat. Phys. **17**, 105 (2021).

[14] Thomas Vojta, Disorder in quantum many-body systems, Annu. Rev. Condens. Matter Phys. **10**, 23 (2019).

[15] Laurent Sanchez-Palencia and Maciej Lewenstein, Disordered quantum gases under control, Nat. Phys. **6**, 87 (2010).

[16] J. E. Lye, L. Fallani, M. Modugno, D. S. Wiersma, C. Fort, and M. Inguscio, Bose-Einstein Condensate in a Random Potential, Phys. Rev. Lett. **95**, 070401 (2005).

[17] D. Clément, A. F. Varón, M. Hugbart, J. A. Retter, P. Bouyer, L. Sanchez-Palencia, D. M. Gangardt, G. V. Shlyapnikov, and A. Aspect, Suppression of Transport of an Interacting Elongated Bose-Einstein Condensate in a Random Potential, Phys. Rev. Lett. **95**, 170409 (2005).

[18] C. Fort, L. Fallani, V. Guarrera, J. E. Lye, M. Modugno, D. S. Wiersma, and M. Inguscio, Effect of Optical

Disorder and Single Defects on the Expansion of a Bose-Einstein Condensate in a One-Dimensional Waveguide, Phys. Rev. Lett. **95**, 170410 (2005).

[19] Giacomo Roati, Chiara D'Errico, Leonardo Fallani, Marco Fattori, Chiara Fort, Matteo Zaccanti, Giovanni Modugno, Michele Modugno, and Massimo Inguscio, Anderson localization of a non-interacting Bose–Einstein condensate, Nature **453**, 895 (2008).

[20] Boris Shapiro, Expansion of a Bose-Einstein Condensate in the Presence of Disorder, Phys. Rev. Lett. **99**, 060602 (2007).

[21] D. Dries, S. E. Pollack, J. M. Hitchcock, and R. G. Hulet, Dissipative transport of a Bose-Einstein condensate, Phys. Rev. A **82**, 033603 (2010).

[22] Yongshan Cheng and S. K. Adhikari, Matter-wave localization in a random potential, Phys. Rev. A **82**, 013631 (2010).

[23] Valentin V. Volchkov, Michael Pasek, Vincent Denechaud, Musawwadah Mukhtar, Alain Aspect, Dominique Delande, and Vincent Josse, Measurement of Spectral Functions of Ultracold Atoms in Disordered Potentials, Phys. Rev. Lett. **120**, 060404 (2018).

[24] Y. Yue, C. A. R. Sá de Melo, and I. B. Spielman, Enhanced transport of spin-orbit-coupled Bose gases in disordered potentials, Phys. Rev. A **102**, 033325 (2020).

[25] Moritz August and José Miguel Hernández-Lobato, Taking gradients through experiments: Lstms and memory proximal policy optimization for black-box quantum control, (2018), CoRR ArXiv:1802.04063.

[26] S. Pilati and Pierbiagio Pieri, Supervised machine learning of ultracold atoms with speckle disorder, Sci. Rep. **9**, 1 (2019).

[27] Shangjie Guo, Amilson R. Fritsch, Craig Greenberg, I. B. Spielman, and Justyna P Zwolak, Machine-learning enhanced dark soliton detection in Bose–Einstein condensates, Machine Learning: Sci. Technol. **2**, 035020 (2021).

[28] Tomi Ohtsuki and Tomohiro Mano, Drawing phase diagrams of random quantum systems by deep learning the wave functions, J. Phys. Soc. Jpn. **89**, 022001 (2020).

[29] N. Saraceni, S. Cantori, and S. Pilati, Scalable neural networks for the efficient learning of disordered quantum systems, Phys. Rev. E **102**, 033301 (2020).

[30] Adriano M. Palmieri, Federico Bianchi, Matteo G. A. Paris, and Claudia Benedetti, Multiclass classification of dephasing channels, Phys. Rev. A **104**, 052412 (2021).

[31] Z. Wu and E. Zaremba, Dissipative Dynamics of a Harmonically Confined Bose-Einstein Condensate, Phys. Rev. Lett. **106**, 165301 (2011).

[32] Sh. Mardonov, M. Modugno, and E. Ya. Sherman, Dynamics of Spin-Orbit Coupled Bose-Einstein Condensates in a Random Potential, Phys. Rev. Lett. **115**, 180402 (2015).

[33] Thibault Scoquart, Thomas Wellens, Dominique Delande, and Nicolas Cherroret, Quench dynamics of a weakly interacting disordered Bose gas in momentum space, Phys. Rev. Res. **2**, 033349 (2020).

[34] Pantita Palittapongarnpim, Peter Wittek, Ehsan Zahedinejad, Shakib Vedaie, and Barry C. Sanders, Learning in quantum control: High-dimensional global optimization for noisy quantum dynamics, Neurocomputing **268**, 116 (2017).

[35] Murphy Niu, Sergio Boixo, Vadim Smelyanskiy, and Hartmut Neven, Universal quantum control through deep reinforcement learning, npj Quantum Inf. **5**, 33 (2019).

[36] A. Bulatov, B. Vugmeister, A. Burin, and H. Rabitz, Nonadiabatic cooling and optimal control in off-resonance dipole optical potentials, Phys. Rev. A **58**, 1346 (1998).

[37] Patrick Doria, Tommaso Calarco, and Simone Montangero, Optimal Control Technique for Many-Body Quantum Dynamics, Phys. Rev. Lett. **106**, 190501 (2011).

[38] J. J. W. H. Sørensen, M. O. Aranburu, T. Heinzel, and J. F. Sherson, Quantum optimal control in a chopped basis: Applications in control of Bose-Einstein condensates, Phys. Rev. A **98**, 022119 (2018).

[39] Bryce M. Henson, Dong K. Shin, Kieran F. Thomas, Jacob A. Ross, Michael R. Hush, Sean S. Hodgman, and Andrew G. Truscott, Approaching the adiabatic timescale with machine learning, Proc. National Acad. Sci. **115**, 13216 (2018).

[40] Marin Bukov, Alexandre G. R. Day, Dries Sels, Phillip Weinberg, Anatoli Polkovnikov, and Pankaj Mehta, Reinforcement Learning in Different Phases of Quantum Control, Phys. Rev. X **8**, 031086 (2018).

[41] Xiao-Ming Zhang, Zezhu Wei, Raza Asad, Xu-Chen Yang, and Xin Wang, When reinforcement learning stands out in quantum control? A comparative study on state preparation, npj Quantum Inf. **5**, 85 (2019).

[42] Mario Krenn, Mehul Malik, Robert Fickler, Radek Lapkiewicz, and Anton Zeilinger, Automated Search for New Quantum Experiments, Phys. Rev. Lett. **116**, 090405 (2016).

[43] Xi Chen, A. Ruschhaupt, S. Schmidt, A. del Campo, D. Guéry-Odelin, and J. G. Muga, Fast Optimal Frictionless Atom Cooling in Harmonic Traps: Shortcut to Adiabaticity, Phys. Rev. Lett. **104**, 063002 (2010).

[44] D. Guéry-Odelin, A. Ruschhaupt, A. Kiely, E. Torrontegui, S. Martínez-Garaot, and J. G. Muga, Shortcuts to adiabaticity: Concepts, methods, and applications, Rev. Mod. Phys. **91**, 045001 (2019).

[45] Roie Dann, Ander Tobalina, and Ronnie Kosloff, Shortcut to Equilibration of an Open Quantum System, Phys. Rev. Lett. **122**, 250402 (2019).

[46] Ken Funo, Neill Lambert, Franco Nori, and Christian Flindt, Shortcuts to Adiabatic Pumping in Classical Stochastic Systems, Phys. Rev. Lett. **124**, 150603 (2020).

[47] M. J. A. Schuetz, J. Knörzer, G. Giedke, L. M. K. Vandersypen, M. D. Lukin, and J. I. Cirac, Acoustic Traps and Lattices for Electrons in Semiconductors, Phys. Rev. X **7**, 041019 (2017).

[48] C. Fefferman, S. Mitter, and H. Narayanan, Testing the manifold hypothesis, J. Am. Math. Soc. **29**, 983 (2016).

[49] Kaiming He Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep residual learning for image recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770 (2016).

[50] G. Gauthier, I. Lenton, N. McKay Parry, M. Baker, M. J. Davis, H. Rubinsztein-Dunlop, and T. W. Neely, Direct imaging of a digital-micromirror device for configurable microscopic optical potentials, Optica **3**, 1136 (2016).

[51] Sylvain Hermelin, Shintaro Takada, Michihisa Yamamoto, Seigo Tarucha, Andreas D. Wieck, Laurent Saminadayar,

Christopher Bäuerle, and Tristan Meunier, Electrons surfing on a sound wave as a platform for quantum optics with flying electrons, Nature **477**, 435 (2011).

[52] R. P. G McNeil, M. Kataoka, C. J. B Ford, C. H. W Barnes, D Anderson, G. A. C Jones, I Farrer, and D. A. Ritchie, On-demand single-electron transfer between distant quantum dots, Nature **477**, 439 (2011).

[53] Adam Paszke, *et al.*, Pytorch: in *Advances in Neural Information Processing Systems 32* (Curran Associates, Inc., 2019), p. 8024.

[54] Diederik P. Kingma and Jimmy Ba, Adam: A method for stochastic optimization, (2017), ArXiv:1412.6980.

[55] Paul T. Boggs and Jon W. Tolle, Sequential quadratic programming, Acta Numerica **4**, 1 (1995).

[56] Tim Miller, Explanation in artificial intelligence: Insights from the social sciences, Artif. Intell. **267**, 1 (2019).

[57] Been Kim, Rajiv Khanna, and Oluwasanmi O. Koyejo, in *Advances in Neural Information Processing Systems*, Vol. 29, edited by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Curran Associates, Inc., 2016).

[58] C. Molnar, *Interpretable Machine Learning* (Lulu.com, Morrisville, NC, 2020).