


Using quantum transport networks for classification: A path toward quantum computing for machine learning

Shmuel Lorber¹, Oded Zimron, Inbal Lorena Zak¹, Anat Milo, and Yonatan Dubi^{*}
Department of Chemistry, Ben-Gurion University of the Negev, Beer Sheva 84105, Israel

 (Received 19 December 2023; revised 3 March 2024; accepted 24 June 2024; published 17 July 2024)

Classification, the computational process of categorizing an input into preexisting classes, is now a cornerstone in modern computation in the era of machine learning. Here, we propose an approach for a quantum physical computer; a quantum classifier, based on quantum transport of particles in a trained quantum network. The classifier is based on sending a quantum particle into a network and measuring the exit point of the particle, which serves as a “class” and can be determined by changing the network parameters, differing from standard quantum computers as no gate operations are required to perform the computation. Using this scheme, we demonstrate three examples of classification. In the first, wave functions are classified according to their overlap with predetermined (random) groups. In the second, we classify wave functions according to their level of localization. Both examples use small training sets and achieve over 95% precision and recall. The third classification scheme is a “real-world problem,” concerning classification of catalytic aromatic aldehyde substrates according to their reactivity. Using experimental data, the quantum classifier reaches an average 86% classification accuracy. We show that the quantum classifier outperforms its classical counterpart for these examples and demonstrates clear advantage, especially in the regime of “small data.” These results pave the way for a classification scheme that can be implemented as an algorithm and potentially realized experimentally on quantum hardware.

DOI: [10.1103/PhysRevApplied.22.014041](https://doi.org/10.1103/PhysRevApplied.22.014041)

I. INTRODUCTION

Recent years have seen a huge advance in computational capabilities that are based on machine-learning (ML) algorithms. Classification, pattern recognition, complex predictions, and adaptive imitation, among others, are typical problems where ML shows incredible advantage over regular search or inference algorithms [1–4]. The growing global use of ML is expected to have substantial energetic impact [5], and the search for alternatives has led to the notion of “computing with physical systems”, namely, going beyond the standard computing paradigm by harnessing natural physical processes for computing [6–10]. Efforts have been devoted to finding hardware elements that directly show nonlinear functionality (e.g., memristors [11–15] or other physical implementations [16–20]).

In spite of the substantial effort devoted to computing with physical systems, to date all current examples underperform compared to standard “silicon” (i.e., regular computers), both in computational performance and in power usage, and an example where a physical computer can do something better than silicon is yet to be found. Here, we propose a quantum approach to supervised classification (SC), based on quantum transport of particles through a

“quantum classification network” (QCN), which in principle can be implemented in current experiments [21]. This quantum physical computer does not require qubits or gates as in “standard” quantum computing. Importantly, we demonstrate that this quantum physical computer outperforms standard ML algorithms in the regime of “small data” (i.e., when the training sets are very small).

SC is one of the most basic computational tasks performed by ML algorithms [2]. We provide three fundamental examples for classification using the QCN. The first is based on classifying wave functions according to their overlap with predetermined groups. The second is based on classifying wave functions according to their level of localization. The third example, based on experimentally measured data, is classification of substrates according to their reactivity properties. We show that for these examples, the quantum-network based SC outperforms standard ML algorithms for small data sets. Moreover, we show that the quantum coherent version also outperforms a similar “classical” computation, thus demonstrating a clear advantage of our computing scheme. Our proposal can be implemented experimentally on various systems such as networks of quantum dots or quantum waveguide arrays (or even “standard” quantum computers), thus paving the way for a design for hardware-based quantum computations.

^{*}Contact author: jdubi@bgu.ac.il

II. SETUP AND FORMULATION

A. Classification protocol

We start by describing the general classification protocol using quantum transport, namely, the problem of assigning an input vector Ψ to a specific class $C[\Psi]$. The vectors belong to a Hilbert space of size L (e.g., $L = 2$ for qubits) and there are N_c classes to choose from. Our classification protocol is defined as follows. A network is constructed that has L entry nodes, a “hidden layer” of M nodes, and a layer of N_c exit nodes, corresponding to N_c classes (see Fig. 1). The network is excited such that the excitation is defined by the state Ψ (i.e., a quantum particle enters the network through the source sites in a way that is defined by the state Ψ , as described in Sec. II B). As a result, current flows through the network and exits through the drain sites. From the currents, the class of Ψ , $C[\Psi]$ is determined by simply evaluating the currents from the different drain sites and assigning the class to the node from which current is maximal. We note that while the size of the source layer L and the size of the exit layer N_c are defined by the problem (i.e., the size of the input vectors and the number of classes, respectively), the size of the “hidden layer” M is a parametric choice, which is determined by trial and error (see, e.g., Fig. 6).

To give a concrete example (as in Fig. 1), consider states from an $L = 3$ Hilbert space, which can be classified into two classes. A specific state Ψ is encoded into the injection of particles to the network from the source (blue circles in Fig. 1) and current leaves from the two drain sites (orange circles in Fig. 1). If most of the current comes out of site 1 (say, the top site), then the state Ψ will be classified as belonging to class 1 (i.e., $C[\Psi] = 1$) and vice versa.

B. The quantum network

We model the QCN using a tight-binding Hamiltonian of the form $\mathcal{H} = \sum_{i,j} h_{ij} c_i^\dagger c_j + h.c.$, where $c_i^\dagger (c_i)$ creates

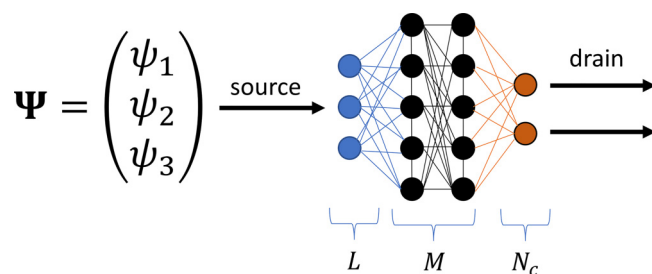


FIG. 1. Schematic representation of the quantum classification network. A wave function Ψ of length L ($L = 3$ in this example) is encoded into an input excitation of particles into the network, the parameters of which are determined following a training protocol. The particles exit the network through N_c sites, representing the N_c classes. The class of Ψ is then determined by the exit site from which the current is maximal.

(annihilates) a particle on the network node i . This is a general approach that can be used to describe many quantum transport networks, including (but not limited to) electron transport in quantum dots [22], exciton transport in bionetworks [23], and photons in waveguides [24,25]. We distinguish between nodes on the entry sites (indexed $i_{\text{in}} = 1, \dots, L$), on the network (“hidden”) layer (indexed $i = 1, \dots, M$) and exit sites (indexed $i_{\text{out}} = 1, \dots, N_c$). The network structure is such that entry sites are coupled to all network sites but not to each other and not directly to the exit sites (and, similarly, the exit sites are only connected to the network sites). The network sites may be interconnected between themselves.

Current propagation through the network, as well as the connection of the network to the environment (i.e., input and output nodes) is modeled using the Gorini-Kossakowski-Sudarshan-Lindblad (GKSL) quantum master equation [26,27] for the density matrix of the system, ρ ,

$$\begin{aligned} \dot{\rho} &= -i[\mathcal{H}, \rho] + \Sigma_k \left(V_k^\dagger \rho V_k - \frac{1}{2}(V_k^\dagger V_k \rho + \rho V_k^\dagger V_k) \right) \\ &= -i[\mathcal{H}, \rho] + \mathcal{L}[\rho], \end{aligned} \quad (1)$$

where $[\cdot, \cdot]$ is the commutator and the V_k are the so-called Lindblad operators. In the above setup, $k = [1, 2]$ for V_{in} (V_{out}), describing the source and drain terms.

The drain term, V_{out} , is defined by a set of N_c V - operators describing the extraction of current from the network, of the form $V_{\text{out},n} = \gamma^{1/2} c_r$ [28], where c_r annihilates a particle on the site r , in which $r = 1, \dots, N_c$ are indices of the exit sites. The source term V_{in} is used to encode the wave functions that are classified. If a vector Ψ is considered, the encoding is performed by setting $V_{\text{in}} = \gamma_{\text{in}}^{1/2} \sum_{i=1}^L \Psi(i) c_i^\dagger$, where the index $i = 1, \dots, L$ runs over the source sites.

The specific value of the rates γ is generally chosen to be $\gamma = 1$ and it has no qualitative effect on the calculation as long as the γ s are multiplied by the same factor (since the currents are proportional to γ). However, the above is not unambiguous, and in some instances (e.g., the second example; see Sec. III B) we have found that varying γ_{out} leads to faster convergence and better classification results.

Once the Hamiltonian is defined (for the training protocol, see Sec. II C) and the source and drain Lindbladians are provided, we proceed by evaluating the current from each exit node [28] (for details, see Appendix A).

We point out that the scheme presented in Secs. II A and II B is different from the one proposed in Ref. [29]. Specifically, in Ref. [29] the system requires two forms of transfer of particles between sites, one coherent (manifested through a tight-binding Hamiltonian) and an additional incoherent transfer, encoded in V operators of the form $\gamma_{ij} c_i^\dagger c_j$, where the γ_{ij} represent incoherent transfer rates between sites. To achieve classification, the authors

thus require optimization of both the Hamiltonian parameters and the γ_{ij} , which we do not. This also casts doubt on the possibility of an experimental realization of the scheme presented in Ref. [29], because experimental control of the incoherent transfer rates, which are determined by an interplay between dephasing, local energies, and other microscopic parameters, is essentially impossible, as it involves a local measurement after a stochastic quantum random walk and requires local control of both Hamiltonian parameters and local dephasing.

C. Network training and validation

In order to succeed in classification, the network parameters—in this case, the tight-binding amplitudes in the Hamiltonian—need to be determined according to a given training set. A training set

$$\text{TS} = \{\varphi_n, C_n\}, \quad n = 1, \dots, N_{\text{TS}}$$

is a set of input vectors φ_n along with their classification C_n , where C_n is an integer from 1 to N_c , referring to the class of the state φ_n , and $n = 1, \dots, N_{\text{TS}}$ is an index running on all the vector states in the training set.

Once the training set is chosen, the next step is the system training, i.e., finding a Hamiltonian that can achieve the classification described above. According to the classification protocol, we define $J_r[\varphi]$ as the current output from the drain site $r = 1, \dots, N_c$ when the input vector is φ . The *class* of φ is thus defined as

$$\mathcal{C}[\varphi] = \text{Index}(\text{Max}(J_r[\varphi], \quad r = 1, \dots, N_c)), \quad (2)$$

i.e., the site index of the drain site with maximal current.

Training the Hamiltonian thus amounts to minimizing the cost function

$$\text{CF}(\mathcal{H}, \text{TS}) = \sum_{n=1}^{N_{\text{TS}}} (\mathcal{C}[\varphi_n] - C_n)^2. \quad (3)$$

For the optimization, we typically use particle-swarm optimization (PSO) algorithms [30], in combination with standard gradient-descent algorithms, which have been found to yield the fastest and most stable convergence.

Once the training is performed, the QCN is validated by applying the algorithm to N_v validation vectors, ψ_n , ($n = 1, \dots, N_v$), which do not belong to the training set, although their “true” class, \mathcal{C}_n can be calculated (or is known externally). Then, by comparing \mathcal{C}_n to the output class of the algorithm $\mathcal{C}[\psi_n]$, the performance of the QCN can be characterized by the standard measures of classification, namely, the *precision* P and *recall* R [2] (perfect precision, $P = 1$, means that in the classification process there were no false positives and perfect recall, $R = 1$, means there were no false negatives).

We note on the fly that “training,” in the way described above, is not the same as the training defined in modern ML algorithms. Indeed, in our case, the full set of data is *a priori* required and “training” implies optimizing the cost function with respect to the given data (rather than updating the system parameters when new data arrive, as in modern ML algorithms). In that sense, our protocol is closer to Hebbian learning of a Hopfield network [31].

III. RESULTS

A. Example I: Group-overlap classification

Our first example entails SC according to the overlap of an input wave function with different sets of predetermined groups of known wave functions. Consider G groups of wave functions, each group containing N_G wave functions φ_n^g , $g = 1, \dots, G$, $n = 1, \dots, N_G$. The class of an input vector ψ will be determined according to its overlap with the wave functions of the group n . Put in mathematical form, we define the average overlap of ψ with the group g ,

$$\eta_g = \frac{1}{N_G} \sum_{n=1}^{N_G} |\langle \varphi_n^g | \psi \rangle|^2. \quad (4)$$

The class of ψ is defined as the index of the group of maximal overlap, $\mathcal{C}[\psi] = \text{Index}[\text{Max}\{\eta_1, \eta_2, \dots, \eta_G\}]$. The quantum network classification is obtained by setting the network to have G exit nodes and training the network such that when a vector φ_n^g is input, most of the current will leave through exit node g , i.e., by minimizing the cost function

$$\text{CF}(\mathcal{H}, \text{TS}) = \sum_g \sum_n \sum_i (J_i[\varphi_n^g] - \delta_{i,g})^2, \quad (5)$$

where i is the index of the exit node and δ is the Kronecker delta function.

We start our demonstration of QCN by examining the simplest possible case of $L = 2$, $G = 2$, and $N_G = 2$, namely, by classifying inputs (of length 2, i.e., qubits) according to their overlap with only two vectors. The training set of the two vectors, $\varphi_{1,2}$, is given by

$$\text{TS} = \begin{cases} \varphi_1 = \cos(x)|0\rangle + \sin(x)|1\rangle, & C_1 = 1, \\ \varphi_2 = \sin(x)|0\rangle + \cos(x)|1\rangle, & C_2 = 2, \end{cases} \quad (6)$$

where we have chosen $x = 0.3$. A network of 2-4-2 is chosen (namely, four nodes in the “hidden layer,” $M = 4$) and trained (by varying the hopping matrix elements such that the cost function is minimized) using PSO as mentioned above.

Once the training is complete, we test the classification using a validation set containing 1000 random states

ψ_n , each assigned a class according to its overlap in the following way. Defining the overlap $\eta_g^{(n)} = \langle \psi_n | \varphi_g \rangle$, the class of ψ_n is $\mathcal{C}_n = 1$ (or 2) if $\eta_1^{(n)} > \eta_2^{(n)}$ (or $\eta_1^{(n)} < \eta_2^{(n)}$). The class $\mathcal{C}[\psi_n]$ defined by the QCN is the index of the exit site which carries the most current when φ_n is inserted into the network.

For this example, we find $P_2 = R_2 = 1$, i.e., *perfect* classification. Put simply, there are no validation vectors ψ_n which belong to class 1 and, when injected to the trained network, have the most current come out of node 2 (and vice versa). This is depicted in Fig. 2(a). For simplicity, we define for each ψ_n an overlap balance $\tilde{\eta}_n = (\eta_1^{(n)} - \eta_2^{(n)}) / (\eta_1^{(n)} + \eta_2^{(n)})$ and a current balance $\tilde{J}_n = (J_1[\psi_n] - J_2[\psi_n]) / (J_1[\psi_n] + J_2[\psi_n])$. Both $\tilde{\eta}_n$ and \tilde{J}_n vary between $\pm \frac{1}{2}$ and perfect classification means that they have the same sign for every ψ_n . In Fig. 2(a), we plot \tilde{J}_n versus $\tilde{\eta}_n$ for all the validation vectors. Indeed, for all of the ψ_n ,

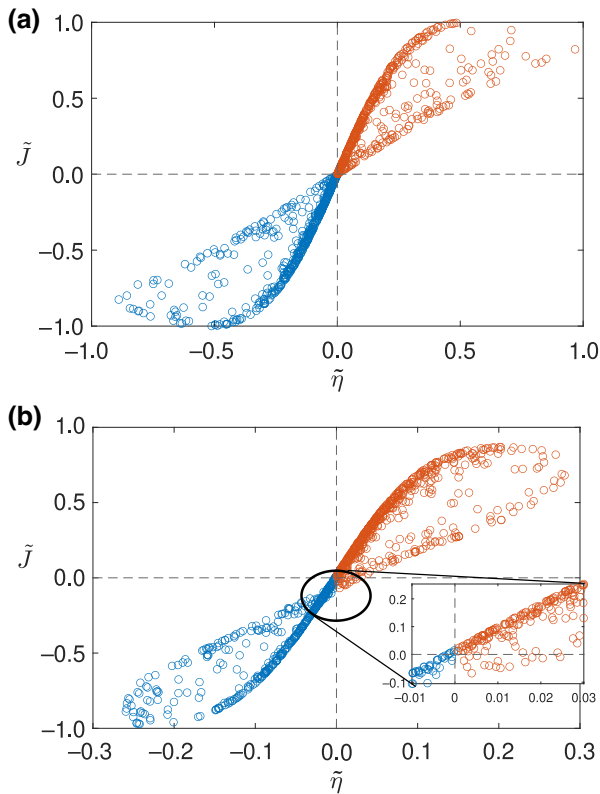


FIG. 2. Performance of the QCN protocol for group-overlap classification. (a) Current balance \tilde{J} versus the overlap balance $\tilde{\eta}$ for $N_v = 1000$ validation vectors for $G = 2$ and $N_G = 2$. No points reside on the second and fourth quarters, indicating that \tilde{J} and $\tilde{\eta}$ always have the same sign. This means perfect classification for this case, with $P = R = 1$. (b) Same as (a) for $G = 2$ and $N_G = 40$ (a training set consisting of two groups with 20 vectors each). We find $P = 0.9914$ and $R = 0.9745$, almost perfect classification. The outliers are due to validation vectors that have very close overlap to the two classes and thus lie close to the $\tilde{\eta} = 0$ point (see the enlargement of this area in the inset).

the sign of \tilde{J} and $\tilde{\eta}$ is the same. Similar results are obtained for other values of x . Specifically, for $x = 0.6$ (representing an overlap > 0.9 between φ_1 and φ_2), we still find precision and recall larger than %98.5, which can be increased with an increasing training time (see Appendix B).

Of course, this is an extremely simple example. To generalize it, we have extended the study to the cases of $N_G = 20$ and $N_G = 40$ (still with $G = 2$, i.e., two classes each containing N_G vectors). The states are of the same form, $\varphi_i = \cos(x_i)|0\rangle + \sin(x_i)|1\rangle$, where the x_i are chosen randomly from a uniform distribution $U[0, 2\pi]$. Using the same training protocol as for the $N_G = 2$, we find $P_{20} = 0.9961$, $R_{20} = 0.9644$ and $P_{40} = 0.96767$, $R_{40} = 0.9984$, i.e., almost perfect classification. The current balance \tilde{J} versus overlap balance $\tilde{\eta}$ for $N_G = 20$ is shown in Fig. 2(b). The outliers, namely, validation states that give a false positive or negative, are such that there is an extremely similar overlap with the two training sets and thus they lie close to the $\tilde{\eta} = 0$ point, as shown in the inset to Fig. 2(b). The specific shape of the plot arises due to the specific choice of vectors in the overlap classes [see Eq. (6)]. Similar tests for other random choices of vectors give similar results.

To see how the QCN compares to a classical classification network (CCN), we have run the same classification problem with a standard classical neural network (for details, see Appendix B). While for $N_G = 2$ the classical algorithm performs perfectly, it provides rather poor classification for $N_G = 20, 40$. In Fig. 3, we plot P and R for $N_G = 2, 20, 40$ for the QCN (stars) and the classical algorithm (circles). The solid and dashed lines are guides to the eye. It is clearly visible that the QCN substantially outperforms the CCN in this regime. However, as we go to much larger training sets (e.g., $N_G = 600$, not shown), the classical algorithm gives $P = 0.9853$ and $R = 1$. This affirms that the advantage we observe is limited to small training sets. It is also important to note that we have allowed the “metaparameters” of the CCN to change; i.e., attempts to adjust the number of nodes or the activation functions has not increased the results for the CCN and the results that we report here are the optimal ones that we have obtained.

To further elaborate on the advantage of the QCN, in Fig. 3(b) the cost function is plotted versus the number of iterations (the so-called calculation epoch) of the training and validation sets for the QCN (red solid and dashed lines, respectively) and the CCN (blue solid and dashed lines, respectively), for $N_G = 40$. For the QCN, the validation set follows the training-set behavior, which is what is expected from a well-performing classification network. For the classical network, on the other hand, the validation set begins to diverge, leading to poor classification. This behavior is a hallmark for the so-called “overfitting” problem [32,33]. It thus seems that the QCN overcomes the overfitting problem for small data sets. Here, we raise a conjecture that for this problem, finding the minimum of

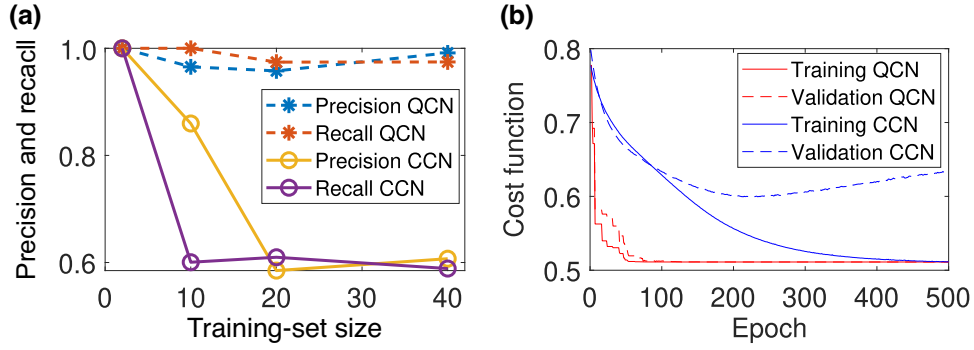


FIG. 3. Advantage of the QCN demonstrated on overlap-based classification. (a) Precision and recall with varying sizes of training sets of the QCN (stars) and CCN (circles). As can be seen, except for $N_{\text{TS}} = 2$, the QCN algorithm shows advantage over the classical algorithm. (b) Cost function versus the training epoch of the training and validation sets (solid and dashed lines, respectively) for the QCN and CCN (red and blue, respectively), for an $N_G = 40$ overlap classification. While for the quantum network the validation-set cost function follows the training-set cost function, the classical-validation cost function diverges, a clear indication of “overfitting” due to the small training set.

the cost function on a (quantum) Hilbert space is a convex optimization problem but that it becomes nonconvex for the classical problem, which leads to the overfitting. Proof of this conjecture is left to future studies.

As a final example for overlap classification, we have performed the calculation for $G = 4$, namely four classification classes. We use a 3-5-4 network. The results for the precision and recall (averaged over the four classes) are given in Table I. Comparing to the classical algorithm, we again find substantial advantage for the QCN.

We point out that this example, although simple to understand, is a somewhat atypical classification problem. The reason is that in this example, the classes and the training set are defined by the same group of input vectors. We thus proceed with a more standard example in Sec. II B.

B. Example II: Classification by level of localization

The next classification assignment that the QCN has been trained to achieve is classification based on the level of localization. We measure the level of localization by the inverse participation ratio (IPR) [34–36], which is a well-known measure of localization, defined by

$$\mathcal{I}[\psi] = \left(\sum_{i=1}^L |\psi(i)|^4 \right)^{-1}. \quad (7)$$

TABLE I. Classification success values for $G = 4$.

	$N_G = 4$	$N_G = 20$	$N_G = 40$
Precision QCN	0.9309	0.9116	0.9383
Recall QCN	0.9000	0.8888	0.8646
Precision CCN	0.5002	0.6537	0.9100
Recall CCN	0.2874	0.6848	0.6533

The IPR ranges from $\mathcal{I} = 1$ for a completely localized wave function to $\mathcal{I} = L$ for a fully delocalized wave function (where the weight is equal over all basis states). The use of different measures of localization (e.g., wave-function entropy) yields similar results.

As a first example, we set $L = 5$ and randomly chose $N_{\text{TS}} = 10$ states, divided into $G = 2$ groups of “localized states” with $\mathcal{I} < 2$ and “extended states” with $\mathcal{I} > 3$ (states with $2 < \mathcal{I} < 3$ have been discarded). After training a 5-8-2 network, 1000 validation wave functions have been tested.

In Fig. 4, we plot the current from exit nodes 1 (red circles) and 2 (blue circles) as a function of the IPR of the test wave function. It can be clearly seen that for validation states with $\mathcal{I} < 3$, most of the current flows out of node 2 and vice versa, as is indeed required for this classification. The precision and recall for this case study have been found to be $P = 0.9409$ and $R = 0.9330$.

As for the case of overlap classification, we proceed to compare the performance of the QCN versus the classical algorithm for different numbers of training-set vectors N_{TS} . As seen in Fig. 4(b), where the precision and recall are plotted as a function of N_{TS} for the QCN (stars) and classical algorithm (circles), the QCN consistently outperforms the classical algorithm over the entire range of examined N_{TS} .

C. Robustness against dephasing

Dephasing, or the general loss of wave-function coherence, may destroy the ability of the network to perform classification, since the classification depends on the coherent transport of the excitations through the trained network. To explore the effect of dephasing, one can add Zeno-type local dephasing terms to the Lindbladian [37] with the V operators $V_i = (\Gamma_{\text{dep}})^{1/2} c_i^\dagger c_i$, where i runs over all network sites and Γ_{dep} is the dephasing rate. In Fig. 5,

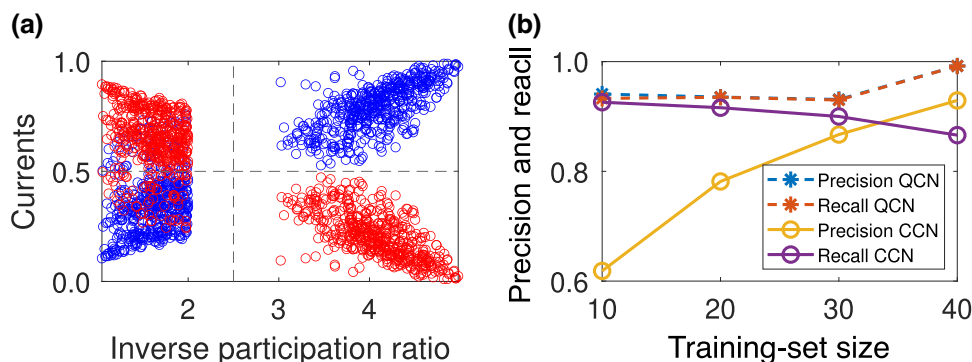


FIG. 4. Performance and advantage of localization-based classification. (a) Currents from exit nodes 1 (red circles) and 2 (blue circles) for 1000 wave functions, plotted versus the wave functions IPR, \mathcal{I} . A clear ability to classify between localized ($\mathcal{I} < 2$) and delocalized $\mathcal{I} > 3$ wave functions is seen—for localized states, most of the current exits from node 1, and for delocalized states, most of the current exits from site 2. (b) Comparison of the precision and recall between quantum and classical network classification, based on the wave-function level of localization (via the IPR; see text) with varying sizes of N_{TS} . The data clearly show the advantage of the QCN.

we plot the precision and recall for a IPR classification for $N_{\text{TS}} = 30$ (the same calculation as in Fig. 2) as a function of the dephasing rate Γ (where Γ is taken in units of the average hopping matrix element of the Hamiltonian, \bar{t}). We find that while the precision actually increases with the dephasing rate, the recall decreases substantially, at $\Gamma \sim 100\bar{t}$. One can evaluate this rate for realistic systems. Noting that $\bar{t} \sim 1 \mu\text{eV}$ for quantum dots [22] and $\bar{t} \sim 1 \text{eV}$ for photonic mazes [25], it follows that the dephasing time should be longer than approximately 50 ns for quantum dots and approximately 50 fs for photonic mazes, which is well within reach in current experimental setups. Similar calculations with different parameters (e.g., different G and N_{TS}) give similar results.

D. Implementation of the QCN to chemical data

The two previous examples, while representative, are somewhat artificial—they have been created to test the

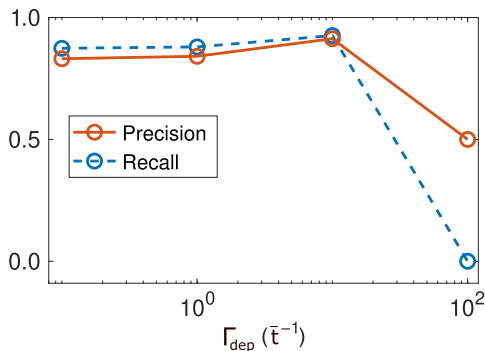


FIG. 5. Robustness against dephasing. Precision and recall of the group localization classification for $N_{\text{TS}} = 30$ as a function of the dephasing rate.

QCN algorithm. Next, we demonstrate the action of the QCN on real experimental data. For this, we turn to the world of chemistry and catalysis, where ML algorithms have been used for many years and yet many problems are characterized by having a small data set, i.e., they are “small-data” problems [38,39].

Specifically, we consider the classification of aromatic aldehyde substrates, based on their preferred reaction conditions to a deuteration reaction. The aldehydes are classified, on a scaling of their reactivity, into three broad categories: (1) those that undergo full deuteration with a catalyst called SIPr but due to their reactivity, tend to proceed and form benzoin—suppressing this further reaction demands also adding boronic acid (BA) to the reaction; (2) aromatic aldehydes that undergo full deuteration with SIPr alone; and (3) those that require a triazolium catalyst (TAC) for their full deuteration. The first group contains the most reactive substrates, while the third group contains the least reactive ones. The substrate reactivity depends on various properties, including steric hindrance, dipole charges, partial charges obtained from natural population analysis (NPA) [40], etc. Since no formal method for determination of the specific reactivity conditions is known, the use of ML algorithms may prove valuable.

In the experiment, 60 substrates (i.e., molecules) have been tested and experimentally classified into one of the three classes described above. We use these data for the classification scheme as follows. Each substrate is characterized by a list of 20 numerical values corresponding to its physical characteristics (some stated above; for the full data set, see Appendix F), which display a broad distribution over the substrates. The values are then normalized such that they are distributed in the range $[-1, 1]$ and then constructed into a “normalized wave function,” namely, a list of normalized parameter values, normalized by itself

to unity. With three classes and ten numerical characteristics, the network thus has ten input nodes (representing the numerical characteristics of the substrate) and $G = 3$ output nodes (representing their reactivity).

We note that since we are working with a very small data set, the typical procedure for classification protocols—namely, dividing the data set into a training set and validation set and determining the performance of the protocol from the chosen training set—may be inadequate. We therefore proceed with a slightly more detailed analysis. We focus on the accuracy, A , defined as the number of correct predictions divided by the total number of predictions [2], rather than the precision and the recall, because of the small data-set size. For an estimation of the protocol accuracy, we thus repeat this procedure ten times, randomly choosing a validation set and a training set at each repetition, and evaluate the accuracy of the protocol for each realization. This gives us an average accuracy and a standard deviation, which is a better statistical description for the protocol performance.

The first step is to determine the optimal network size. For the performance assessment, we follow a version of a random cross-validation procedure. We randomly select from the available data a training set of size $N_{\text{TS}} = 10$ and train the network (i.e., optimize the cost function). The accuracy is then evaluated with a validation set with ten random vectors (different from the training set). The procedure is then repeated 10 times, each time with a different randomly chosen training and validation sets. The final accuracy value received is a mean of the ten calculations. In Fig. 6(a), we plot the accuracy (mean and standard deviation) as a function of the size of the internal layer (e.g., the black circles in Fig. 1). We find that a network with seven sites gives the best accuracy, of approximately 86%.

We then turn to determining how many vectors the training set requires to obtain good classification accuracy. Using the optimal network size found above, in Fig. 6(b) we show the accuracy as a function of the training-set size. Surprisingly, we find that the best accuracy is actually reached for a rather small training-set size $N_{\text{TS}} = 10$ and that increasing the training-set size actually reduces the accuracy on average. Similar results have been obtained for the precision and recall (see Appendix G).

The results of Fig. 6 show that the QCN protocol averages at an accuracy of around $A = 0.86$ and for some validation sets reaches up very close to 100% accuracy (similar values have been obtained for the precision and recall). These values are comparable to (and even higher than) the accuracy of similar calculations performed with classical algorithms [38,41–43] and yet have been obtained with a much smaller training set (of only approximately ten vectors). This demonstrates the possibility of quantum advantage in the QCN protocol in the regime of small data.

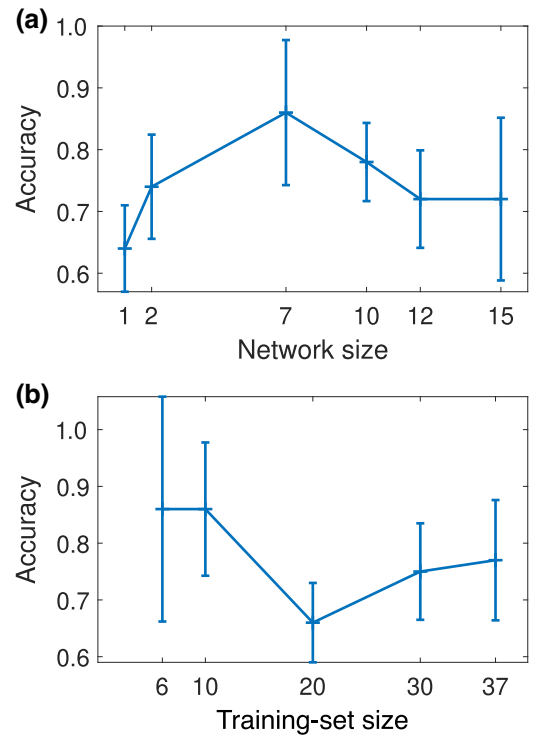


FIG. 6. Performance of the QCN protocol on real-world data. The classification protocol (see text for details) is used for randomly chosen training and validation sets, from which the average and standard deviation of the accuracy are evaluated. (a) Average classification accuracy for a network of constant $N_{\text{TS}} = 10$, as a function of the size M of the intermediate (hidden) layer (see Fig. 1). The network that presents the best performance is a 10-7-3 network. (b) Mean accuracy of the QCN classification for different numbers of training-set vectors, N_{TS} . The best result is obtained for $N_{\text{TS}} = 10$, with accuracy $A = 0.86$, and reaching up to $A = 1.0$ for some validation sets. The fact that the best N_{TS} and M are so small shows the potential of the QCN protocol for small-data problems. The error bars indicate the standard deviation.

IV. SUMMARY AND DISCUSSION

In summary, we have presented a paradigm for a “quantum computer,” namely, a quantum device aimed at performing calculations, based on quantum transport of particles through a network. This approach is completely orthogonal to standard quantum computers, since no gates or qubits are involved, and it is designed specifically to perform supervised classification, a computational task relevant to modern ML.

We examine three examples for classification. The first two examples are based on the overlap of functions with a predefined set and on classification according to the level of wave-function localization. These examples demonstrate a clear advantage over the classical algorithms. The third example aims at classifying aromatic aldehyde substrates according to their chemical reactivity in deuteration

reaction, using measured data, thus demonstrating how the QCN algorithm can be implemented in a real-world classification task with a small data set.

Admittedly, the QCN algorithm that we present is computationally more expensive and, for large system sizes, may be more time consuming than the classical algorithms. However, it shows a clear advantage, which is especially relevant for the regime of small data [44], where the available data sets are limited in size, a regime that seems highly relevant in fields such as material design and chemistry [38,45–47]. This may become practical for situations in which calculation time is an unimportant factor but recall and precision are important.

What is the origin of the advantage of the QCN algorithm? One possible conjecture is as follows. The cost function of the QCN algorithm is based on the relation between the currents and the hopping matrix elements in a network, a relation that is typically of a Lorentzian nature. This means that the optimization landscape is fairly shallow, i.e., the local minima are not “exponentially deep” but are “Lorentzianly deep,” which allows the algorithm to effectively escape local minima and to reach the global minimum even for a small training set. If this is the case, how is it that this works for such small training sets? It may also be the case that within the optimization landscape, the local minima are separated by plateaus, in some form of quantum concentration of measure [48]. These hypotheses are left for future investigations.

Importantly, the QCN scheme that we present here is not just an algorithm to be implemented on a computer. Rather, it is a scheme that can, in principle, be implemented using current state-of-the-art experiments. One possible implementation could be through using photonic networks (see, e.g., Refs. [21,49–51]), which are arrays of vertically grown waveguides, the position and size of which (and hence the coupling between them) can be controlled and in which single-photon states can be generated. Another possible way to implement the QCN algorithm is by using existing quantum computers, by exciting a coherent superposition of single-excited qubits encoding the input state and measuring some qubits defined as “exit qubits” (of course, multiple measurements will be required). While this is, in a way, an under-use of such systems (because only a small submanifold of the full available Hilbert space is used), as long as the coupling between the qubits can be tuned, the QCN algorithm can be demonstrated (without the use of any operations during computation). A third possibility is to use coherent manipulation and preparation of states in arrays of semiconductor quantum dots [52–56]. Yet a fourth possibility—at least similar in spirit—is to use coherent light in photonic arrays, where the refraction index of each pixel in the array can be controlled and tuned [8,57] and the propagation of light (governed by Maxwell’s equations) mimics the coherent propagation of quantum particles in the QCN. One possibility

to consider is that the efficiency of the QCN is not due to quantum correlations but, rather, to coherent propagation (which can be achieved by using coherent light). Investigations in all of these directions are ongoing. Thus, this work hopefully paves the way for a quantum machine aimed at performing specific ML computational tasks, beyond the standard paradigm of quantum computing.

The custom code that has been created during the work that has led to the main results of this paper is published in a public GitHub repository [58].

APPENDIX A: CURRENT FROM THE LINDBLAD EQUATION

As discussed in Sec. II, the system is described by a general tight-binding Hamiltonian and the Lindblad equation,

$$\begin{aligned}\dot{\rho} &= -i[\mathcal{H}, \rho] + \Sigma_k \left(V_k^\dagger \rho V_k - \frac{1}{2}(V_k^\dagger V_k \rho + \rho V_k^\dagger V_k) \right) \\ &= -i[\mathcal{H}, \rho] + \mathcal{L}[\rho].\end{aligned}$$

We limit the calculation to a single-exciton manifold (we expect only minor changes if one uses the full Fock space, as demonstrated in Ref. [28]).

We solve the Lindblad equation and evaluate the current at the steady state. i.e., $\dot{\rho}_s = 0$ [23,59–61].

Once ρ_s is at hand, we evaluate the current, J , from the extraction sites $J_i = d\langle n_i \rangle / dt$, where $\langle n_i \rangle$ is the on-site density (i is the site identifier), via the relation $\langle n_i \rangle = \text{Tr}(\hat{n}_i \rho)$. Substituting the expression for the on-site density into J_i gives an equation for the current,

$$J_{ij} = \frac{d\langle n_i \rangle}{dt} = \frac{d}{dt} \text{Tr}(\hat{n}_i \rho). \quad (\text{A1})$$

Inserting the derivative into the trace and applying it on its contents yields

$$J_i = \text{Tr}(\dot{\hat{n}}_i \rho_s + \hat{n}_i \dot{\rho}_s). \quad (\text{A2})$$

Considering only exit sites, this expression formally becomes

$$J_{\text{ext}} = \text{Tr}(\hat{n}_{\text{ext}}(-i[\mathcal{H}, \rho_s] + \mathcal{L}_{\text{ext}}[\rho_s])) = 0. \quad (\text{A3})$$

The above expression vanishes due to trivial current conservation at the steady state (the two terms are identical in magnitude and opposite in sign). In order to evaluate the exiting current, we thus evaluate only the term relating to the exit sites (or, more specifically, to the measuring apparatus), e.g., the term

$$J_{\text{ext}} = \text{Tr}(\hat{n}_{\text{ext}} \mathcal{L}_{\text{ext}}[\rho_s]). \quad (\text{A4})$$

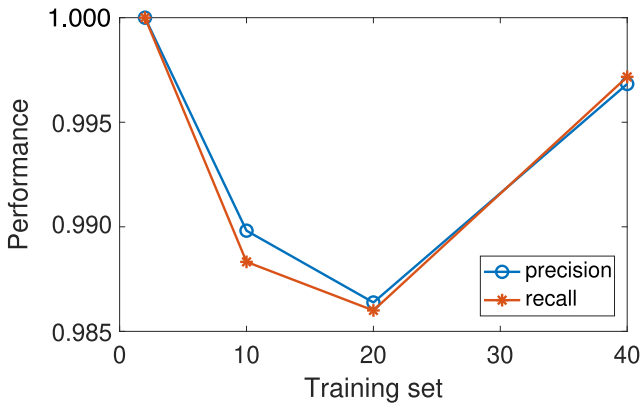


FIG. 7. Recall and precision of the overlap task with $x = 0.6$, for different values of N_G .

APPENDIX B: TRAINING THE CLASSICAL CLASSIFICATION NETWORK

In order to compare the QCN and the classical network, we use the following procedure. First, we take exactly the same training set for both cases. Then, we train the classical network using KERAS [62], which is a convenient application programming interface (API) for implementing ML models in PYTHON. The activation function of the CCN is “Relu,” the output-layer activation is a “sigmoid,” the optimizer of the network is “Adam,” and the loss function used is “CROSSENTROPY.”

APPENDIX C: OVERLAP CLASSIFICATION FOR $x = 0.6$

In Fig. 7, we plot the precision and recall for the overlap classification problem with $x = 0.6$, with $N_v = 1000$ validation vectors for $G = 2$ and different values of the training-set size [compare these results with those of Fig. 3(a)]. Specifically, for the training-set size $N_G = 2$, we choose vectors as in Eq. (6), with $x = 0.6$. For training sets larger than 2, we chose vectors randomly in the form $\varphi_i = \cos(x_i)|0\rangle + \sin(x_i)|1\rangle$ [similar to Fig. 3(a)], where the x_i have been chosen randomly from a uniform distribution $U[0, 0.6]$, thus generating two groups with much larger overlap between them. Even in this case, the precision and recall that we obtain are larger than 0.985.

Even for $x = 0.6$, representing an overlap of $\langle \varphi_1 | \varphi_2 \rangle \sim 0.932$, the calculation yields a classification recall and precision above 0.985.

APPENDIX D: OVERFITTING

In classical ML, overfitting is typically attributed to one of three causes: (i) high hypothesis complexity, (ii) flawed iteration propagation or inappropriate cost function, or (iii) a noisy or nonrepresentative data set [32,33]. In our examples, one can rule out the two first possibilities and thus it seems that the classical algorithm fails due to the fact that the training set is not large enough and is thus nonrepresentative of the full range of outcomes. The quantum algorithm that we present seems to overcome this problem.

APPENDIX E: ROLE OF DEPHASING

The dephasing has been added to the Lindblad equation with the Lindblad operator $L_{\text{dep}} = \sqrt{\gamma_{\text{dep}}} \sum_i a_i^\dagger a_i$, while a_i^\dagger and a_i are the creation and annihilation operators. The calculation of precision and recall in the presence of dephasing has been done for the overlap classification problem, for $G = 2, N_G = 2, 10, 20$, and for the IPR classification problem, for $N_{\text{TS}} = 10, 20, 30, 40$. The recall and precision as a function of the dephasing rate γ_{dep} , for different values of N_G are shown in Figs. 8 and 9. We find that the robustness against dephasing diverges. While the overlap-based classifiers Fig. 8 show good to strong robustness (between $\tau = 1$ and $\tau = 10^{-4}$, where τ is the dephasing time of the system), the IPR classifiers show typically less robust behavior, as in the case of $N_{\text{TS}} = 10$, where in a dephasing time of less than $\tau = 10^4$, the performance of the network starts to decay rapidly. Yet, as the training-set size increases, the reaction to dephasing becomes more stable, as seen in Fig. 9.

We point out that the units of γ_{dep} and τ are taken in relation to the units of the average hopping elements of the Hamiltonian, \bar{t} . One can evaluate this rate for realistic systems. Noting that $\bar{t} \sim 1 \mu\text{eV}$ for quantum dots [22] and $\bar{t} \sim 1 \text{eV}$ for photonic mazes [25], it follows that the dephasing time should be longer than approximately 0.3 ns for quantum dots and approximately 0.3 fs for photonic mazes, which is well within the reach of current

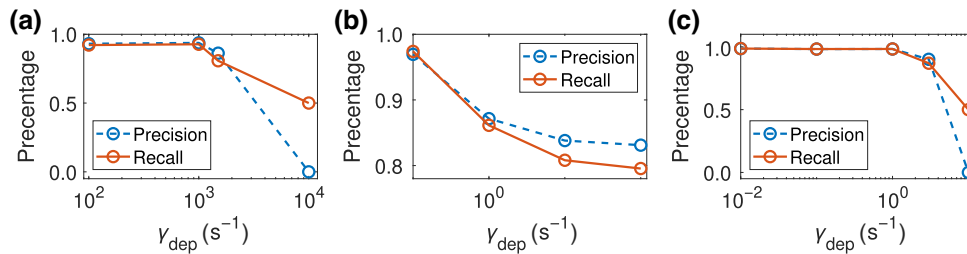


FIG. 8. Recall and precision of the overlap task as a function of the dephasing rate γ_{dep} , for different values of N_G .

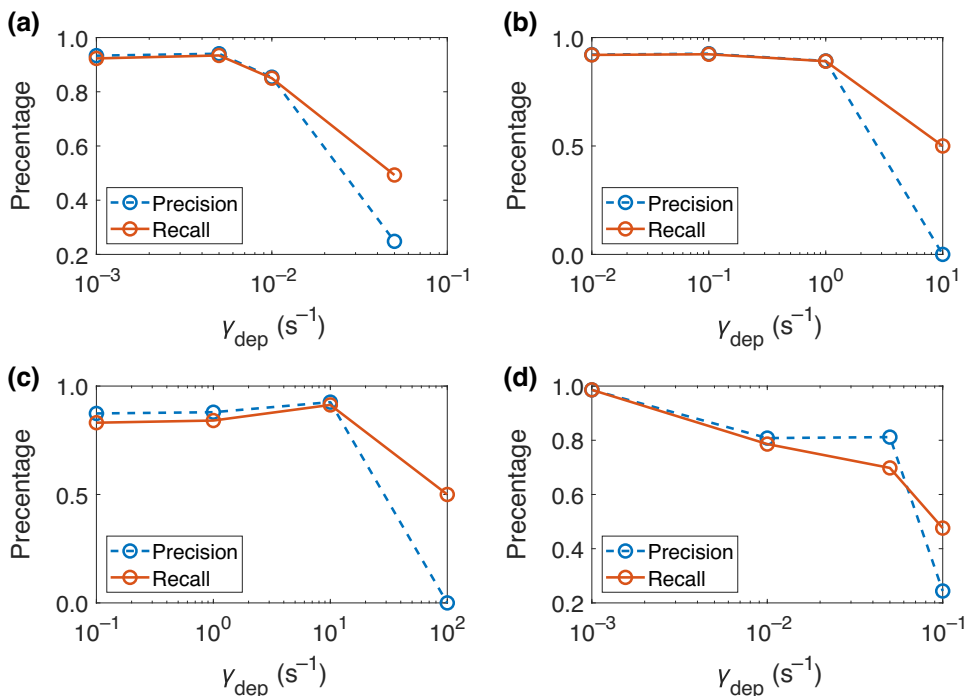


FIG. 9. Recall and precision of the IPR task as a function of the dephasing rate γ_{dep} , for different values of N_{TS} .

experimental setups. Of course, if one wishes to use the QCN as an algorithmic approach to “small-data” problems, dephasing is irrelevant. However, for future physical implementations of the QCN, dephasing needs to be addressed. The limiting dephasing times will depend on the specific implementation and the size of the training set, which can change the dephasing robustness by a few orders of magnitude.

APPENDIX F: IMPLEMENTATION OF QCN ON REAL-WORLD DATA

The QSC distinguishes between substrates based on a collection of 20 parameters, which have been specifically calculated for these purposes. The parameters represent different attributes of the substrates, which tend to lead to classification by the reactivity of each group. The detailed parameters are as follows:

(1) Two vibration-frequency calculations have been made using the GAUSSIAN 16 software [63]. The first parameter is the frequency of vibrations between the carbonyl carbon (“2” in Fig. 10) and the neighboring oxygen (“3” in Fig. 10) and a second parameter is determined as the frequency of vibrations between the carbonyl carbon and the neighboring group (“7” in Fig. 10).

(2) Four dipole calculations have been made using GAUSSIAN. Each substrate has three dipole parameters taken along each axis (see Fig. 10) plus the total dipole.

(3) The distances (bond lengths) between the atoms 2-1, 2-3, and 2-7 (see Fig. 10) have been measured (also using GAUSSIAN).

(4) The partial charges on each atom (1, 2, 3, and 7) obtained from natural population analysis (NPA) [40] have been calculated using the NBO 3.1 extension in GAUSSIAN.

(5) The charge differences between atoms 1-2, 2-3, and 2-7 have been calculated.

(6) Using Verloop’s sterimol program and the open-source PYTHON module STERIMOL [64], three more steric parameters have been obtained in order to catch the

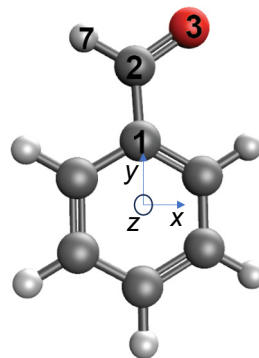


FIG. 10. The coordinate system used for the geometry and frequency parameters. While the y axis is defined as the axis along the aromatic ring carbonyl, the x axis is orthogonal to it in the plane of the ring and the z axis is going into the page. 2, Carbonyl carbon; 1, the carbon of the aromatic ring attached to the carbonyl; 3, the oxygen atom; 7, the group connected to 1.

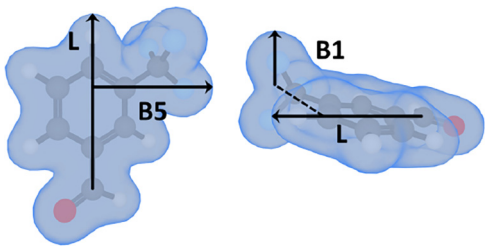


FIG. 11. Verloop's stericol parameters L , $B1$, and $B5$, demonstrated on 2-Fluorobenzaldehyde.

orientation of the aldehyde in space. In order to calculate stericol values, one needs to determine a principal axis: here, the principal axis, L , is the carbonyl-ring axis (see Fig. 11). Thus the first steric parameter calculated, denoted L , is the added length of the substituted ring and the carbonyl ring bond distance. the other two are $B1$, the minimal width perpendicular to the principal axis, and $B5$, the maximal width perpendicular to the principal axis (Fig. 11).

(7) The location of $B5$ on L has been calculated.

APPENDIX G: TRAINING PROCESS ON REAL-WORLD DATA

The construction of a classification network is a trial-and-error process. The first task after choosing the basic model, the QSC, is to find the best optimization technique suited for the problem. At its core, optimization involves

the process of finding the best solution or outcome from a set of possible choices, while adhering to certain constraints or objectives. The convergence speed is an important attribute of optimization methods, with faster methods yielding quicker solutions but potentially missing global optima. With regard to the QSC, the network dimensions (i.e., its input, output, and intermediate sizes) are predetermined manually but the parameters of the Hamiltonian, serving as an adjacency matrix, need to be optimized by the desired optimizer. The best optimization method chosen for this mission has been particle-swarm optimization (PSO) [30], an algorithm inspired by the social behavior of birds and fish. It models potential solutions as particles in a multidimensional space, which move toward the optimal solution by adjusting their positions based on their own experiences and the experiences of their neighbors. It is a fairly heavy optimizer with high function-investigation abilities, although it tends to get stuck in local minima. While using PSO for finding the Hamiltonian parameters, another challenge is to tune the network dimensions, which are manually predetermined as mentioned. In Fig. 6(a), we present the outcomes of six different PSO trains of the network, for six different numbers of intermediate nodes.

Another important feature of the training process is the training set. The training set is a subset of the states to classify, chosen to train the system. The remaining states, or a suitable number out of them, will be used for validating the system. These states are the "validation set." A good training process will use the minimum size of the

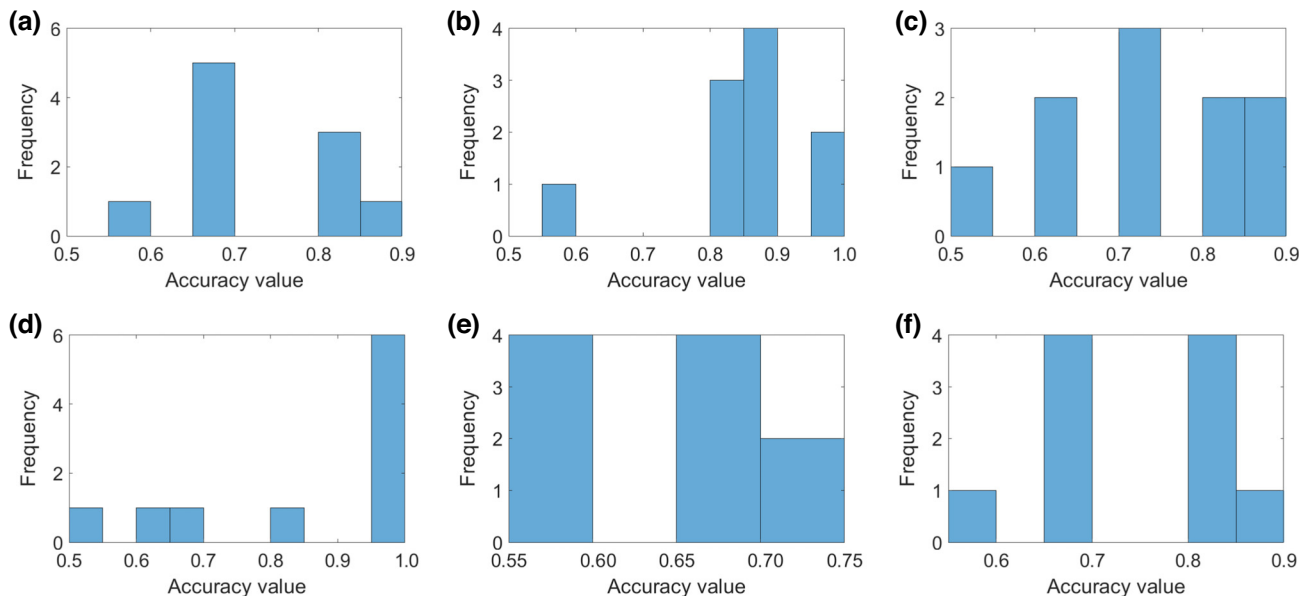


FIG. 12. Histograms describing the validation process leading to Fig. 6. (a)–(c) Different intermediate node numbers (M) with equal-size training sets ($N_{TS} = 10$): (a) $M = 2$; (b) $M = 7$; (c) $M = 15$. (d)–(f) M is fixed and N_{TS} varies: (d) $N_{TS} = 6$; (e) $N_{TS} = 20$; (f) $N_{TS} = 30$. Each plot contains the accuracy outcomes of ten different trains, performed on different random training sets and validated along a validation set of $N_{VS} = 10$ that is also chosen at random. The average of each histogram is then calculated and inserted to produce Fig. 6.

training set and yield maximum classification performance on the validation set, not a trivial task due to the difficulty in making generalizations as the number of examples, the training states, is descending. In Fig. 6(b), we present another five PSO outcomes for five different training-set sizes. The graph shows how, for this problem, the best performance is achieved for a very small training-set size, which points to the high generalization abilities of the QSC.

In Fig. 12, we show histograms aimed to provide a better insight into the training process and how we produced Fig. 6. In Figs. 12(a)–12(c), different dimensions of the network are trained, for the same training-set size $N_{\text{TS}} = 10$. For each network dimension, we perform ten different PSO trains, with different randomly chosen training sets and validation sets. This is called a random tenfold cross-validation. The sizes of the TS and the VS stay at $N = 10$ for each training session. After obtaining the accuracy values from each network classification, the average outcome is calculated and the performance ability of the network is determined. This process of choosing a random validation set instead of just choosing the remaining (not training) states is chosen due to the high dependence of the classification outcome on the specific validation set and on our willingness to compare between different values of N_{TS} in a finite small substrate sample ($N_{\text{total}} = 60$), as can be seen in Figs. 12(d)–12(f). In Figs. 12(d)–12(f), we present histogram plots of three models with equal intermediate node numbers, 2, and varying sizes of training set. The process of producing the histograms is identical to the one described earlier, while the validation-set size remains 10.

-
- [1] J. Alzubi, A. Nayyar, and A. Kumar, Machine learning from theory to algorithms: An overview, *J. Phys.: Conf. Ser.* **1142**, 012012 (2018).
- [2] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms* (Cambridge University Press, New York, NY, 2014).
- [3] C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, New York, NY, 2006).
- [4] X. Gao and L.-M. Duan, Efficient representation of quantum many-body states with deep neural networks, *Nat. Commun.* **8**, 1 (2017).
- [5] A. de Vries, The growing energy footprint of artificial intelligence, *Joule* **7**, 2191 (2023).
- [6] M. Stern and A. Murugan, Learning without neurons in physical systems, *Annu. Rev. Condens. Matter Phys.* **14**, 417 (2023).
- [7] N. Mohseni, P. L. McMahon, and T. Byrnes, Ising machines as hardware solvers of combinatorial optimization problems, *Nat. Rev. Phys.* **4**, 363 (2022).
- [8] P. L. McMahon, The physics of optical computing, *Nat. Rev. Phys.* **5**, 717 (2023).
- [9] G. Wetzstein, A. Ozcan, S. Gigan, S. Fan, D. Englund, M. Soljačić, C. Denz, D. A. Miller, and D. Psaltis, Inference in artificial intelligence with deep optics and photonics, *Nature* **588**, 39 (2020).
- [10] M. Stern, D. Hexner, J. W. Rocks, and A. J. Liu, Supervised learning in physical networks: From machine learning to learning machines, *Phys. Rev. X* **11**, 021045 (2021).
- [11] A. Thomas, Memristor-based neural networks, *J. Phys. D: Appl. Phys.* **46**, 093001 (2013).
- [12] L. Gao, Q. Ren, J. Sun, S.-T. Han, and Y. Zhou, Memristor modeling: Challenges in theories, simulations, and device variability, *J. Mater. Chem. C* **9**, 16859 (2021).
- [13] Z. Wang, C. Li, W. Song, M. Rao, D. Belkin, Y. Li, P. Yan, H. Jiang, P. Lin, M. Hu, *et al.*, Reinforcement learning with analogue memristor arrays, *Nat. Electron.* **2**, 115 (2019).
- [14] W. Xu, J. Wang, and X. Yan, Advances in memristor-based neural networks, *Front. Nanotechnol.* **3**, 645995 (2021).
- [15] A. Mehonic, A. Sebastian, B. Rajendran, O. Simeone, E. Vasilaki, and A. J. Kenyon, Memristors—from in-memory computing, deep learning acceleration, and spiking neural networks to the future of neuromorphic and bio-inspired computing, *Adv. Intell. Syst.* **2**, 2000085 (2020).
- [16] J. Ewaniuk, J. Carolan, B. J. Shastri, and N. Rotenberg, Imperfect quantum photonic neural networks, *Adv. Quantum Technol.* **6**, 2200125 (2023).
- [17] A. d’Arco, F. Xia, A. Boniface, J. Dong, and S. Gigan, Physics-based neural network for non-invasive control of coherent light in scattering media, *Opt. Express* **30**, 30845 (2022).
- [18] R. H. Lee, E. A. Mulder, and J. B. Hopkins, Mechanical neural networks: Architected materials that learn behaviors, *Sci. Rob.* **7**, eabq7278 (2022).
- [19] J. Spall, X. Guo, and A. I. Lvovsky, Hybrid training of optical neural networks, *Optica* **9**, 803 (2022).
- [20] L. G. Wright, T. Onodera, M. M. Stein, T. Wang, D. T. Schachter, Z. Hu, and P. L. McMahon, Deep physical neural networks trained with backpropagation, *Nature* **601**, 549 (2022).
- [21] F. Caruso, A. Crespi, A. G. Ciriolo, F. Sciarrino, and R. Osellame, Fast escape of a quantum walker from an integrated photonic maze, *Nat. Commun.* **7**, 1 (2016).
- [22] S. Sarkar and Y. Dubi, Emergence and dynamical stability of a charge time-crystal in a current-carrying quantum dot simulator, *Nano Lett.* **22**, 4445 (2022).
- [23] E. Zerah Harush and Y. Dubi, Do photosynthetic complexes use quantum coherence to increase their efficiency? Probably not, *Sci. Adv.* **7**, eabc4631 (2021).
- [24] S. Mookherjea and A. Yariv, Optical pulse propagation in the tight-binding approximation, *Opt. Express* **9**, 91 (2001).
- [25] Y. Chen, X. Chen, X. Ren, M. Gong, and G.-C. Guo, Tight-binding model in optical waveguides: Design principle and transferability for simulation of complex photonics networks, *Phys. Rev. A* **104**, 023501 (2021).
- [26] G. Lindblad, On the generators of quantum dynamical semigroups, *Commun. Math. Phys.* **48**, 119 (1976).
- [27] V. Gorini, A. Kossakowski, and E. C. G. Sudarshan, Completely positive dynamical semigroups of N -level systems, *J. Math. Phys.* **17**, 821 (1976).
- [28] E. Zerah-Harush and Y. Dubi, Effects of disorder and interactions in environment assisted quantum transport, *Phys. Rev. Res.* **2**, 023294 (2020).

- [29] L.-J. Wang, J.-Y. Lin, and S. Wu, Implementation of quantum stochastic walks for function approximation, two-dimensional data classification, and sequence classification, *Phys. Rev. Res.* **4**, 023058 (2022).
- [30] R. Poli, J. Kennedy, and T. Blackwell, Particle swarm optimization, *Swarm Intell.* **1**, 33 (2007).
- [31] P. Tolmachev and J. H. Manton, in *2020 International Joint Conference on Neural Networks (IJCNN)*, edited by A. Roy (IEEE, Glasgow, UK, 2020), p. 1.
- [32] T. Dietterich, Overfitting and undercomputing in machine learning, *ACM Comput. Surv. (CSUR)* **27**, 326 (1995).
- [33] X. Ying, An overview of overfitting and its solutions, *J. Phys.: Conf. Ser.* **1168**, 022022 (2019).
- [34] S. Hikami, Localization length and inverse participation ratio of two dimensional electron in the quantized Hall effect, *Prog. Theor. Phys.* **76**, 1210 (1986).
- [35] Y. V. Fyodorov and A. D. Mirlin, Level-to-level fluctuations of the inverse participation ratio in finite quasi 1D disordered systems, *Phys. Rev. Lett.* **71**, 412 (1993).
- [36] F. Evers and A. Mirlin, Fluctuations of the inverse participation ratio at the Anderson transition, *Phys. Rev. Lett.* **84**, 3690 (2000).
- [37] H.-P. Breuer and F. Petruccione, *The Theory of Open Quantum Systems* (Oxford University Press, USA, 2002).
- [38] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh, Machine learning for molecular and materials science, *Nature* **559**, 547 (2018).
- [39] H. Shalit Peleg and A. Milo, Small data can play a big role in chemical discovery, *Angew. Chem.* **135**, e202219070 (2023).
- [40] A. E. Reed, R. B. Weinstock, and F. Weinhold, Natural population analysis, *J. Chem. Phys.* **83**, 735 (1985).
- [41] M. H. Segler and M. P. Waller, Neural-symbolic machine learning for retrosynthesis and reaction prediction, *Chem. A Eur. J.* **23**, 5966 (2017).
- [42] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande, MOLECULENET: A benchmark for molecular machine learning, *Chem. Sci.* **9**, 513 (2018).
- [43] B. Liu, B. Ramsundar, P. Kawthekar, J. Shi, J. Gomes, Q. Luu Nguyen, S. Ho, J. Sloane, P. Wender, and V. Pande, Retrosynthetic reaction prediction using neural sequence-to-sequence models, *ACS Cent. Sci.* **3**, 1103 (2017).
- [44] Z. Cui, Machine learning and small data, *Educ. Meas.: Issues Pract.* **40**, 8 (2021).
- [45] A. L. Liu, R. Venkatesh, M. McBride, E. Reichmanis, J. C. Meredith, and M. A. Grover, Small data machine learning: Classification and prediction of poly (ethylene terephthalate) stabilizers using molecular descriptors, *ACS Appl. Polym. Mater.* **2**, 5592 (2020).
- [46] Y. Haraguchi, Y. Igarashi, H. Imai, and Y. Oaki, Size-distribution control of exfoliated nanosheets assisted by machine learning: Small-data-driven materials science using sparse modeling, *Adv. Theor. Simul.* **4**, 2100158 (2021).
- [47] R. Drechsler, S. Huhn, and C. Plump, in *2020 23rd Euromicro Conference on Digital System Design (DSD)*, edited by A. Skraba (IEEE, Kranj, Slovenia, 2020), p. 518.
- [48] P. Hayden, in *Proceedings of Symposia in Applied Mathematics*, edited by S. J. Lomonaco Jr., Vol. 68 (American Mathematical Society, Washington DC, 2010).
- [49] S. Viciani, S. Gherardini, M. Lima, M. Bellini, and F. Caruso, Disorder and dephasing as control knobs for light transport in optical fiber cavity networks, *Sci. Rep.* **6**, 37791 (2016).
- [50] H. Tang, X.-F. Lin, Z. Feng, J.-Y. Chen, J. Gao, K. Sun, C.-Y. Wang, P.-C. Lai, X.-Y. Xu, Y. Wang, *et al.*, Experimental two-dimensional quantum walk on a photonic chip, *Sci. Adv.* **4**, 3174 (2018).
- [51] D. N. Biggerstaff, R. Heilmann, A. A. Zecevik, M. Gräfe, M. A. Broome, A. Fedrizzi, S. Nolte, A. Szameit, A. G. White, and I. Kassal, Enhancing coherent transport in a photonic network using controllable decoherence, *Nat. Commun.* **7**, 11282 (2016).
- [52] F. Van Riggelen, N. Hendrickx, W. Lawrie, M. Russ, A. Sammak, G. Scappucci, and M. Veldhorst, A two-dimensional array of single-hole quantum dots, *Appl. Phys. Lett.* **118**, 044002 (2021).
- [53] N. W. Hendrickx, W. I. Lawrie, M. Russ, F. van Riggelen, S. L. de Snoo, R. N. Schouten, A. Sammak, G. Scappucci, and M. Veldhorst, A four-qubit germanium quantum processor, *Nature* **591**, 580 (2021).
- [54] S. G. Philips, M. T. Mądzik, S. V. Amitonov, S. L. de Snoo, M. Russ, N. Kalhor, C. Volk, W. I. Lawrie, D. Brousse, L. Tryputen, *et al.*, Universal control of a six-qubit quantum processor in silicon, *Nature* **609**, 919 (2022).
- [55] E. Chanrion, D. J. Niegemann, B. Bertrand, C. Spence, B. Jadot, J. Li, P.-A. Mortemousque, L. Hutin, R. Maurand, X. Jehl, M. Sanquer, S. De Franceschi, C. Bäuerle, F. Balestro, Y.-M. Niquet, M. Vinet, T. Meunier, and M. Urdampilleta, Charge detection in an array of CMOS quantum dots, *Phys. Rev. Appl.* **14**, 024066 (2020).
- [56] W. Lawrie, H. Eenink, N. Hendrickx, J. Boter, L. Petit, S. Amitonov, M. Lodari, B. Paquelet Wuetz, C. Volk, S. Philips, *et al.*, Quantum dot arrays in silicon and germanium, *Appl. Phys. Lett.* **116**, 080501 (2020).
- [57] H. Zhu, J. Zou, H. Zhang, Y. Shi, S. Luo, N. Wang, H. Cai, L. Wan, B. Wang, X. Jiang, *et al.*, Space-efficient optical computing with an integrated chip diffractive neural network, *Nat. Commun.* **13**, 1044 (2022).
- [58] <https://github.com/ShmueLorber/Quantum-Classification>
- [59] C. Guo, K. Wang, E. Zerah-Harush, J. Hamill, B. Wang, Y. Dubi, and B. Xu, Molecular rectifier composed of DNA with high rectification ratio enabled by intercalation, *Nat. Chem.* **8**, 484 (2016).
- [60] E. Zerah-Harush and Y. Dubi, Enhanced thermoelectric performance of hybrid nanoparticle–single-molecule junctions, *Phys. Rev. Appl.* **3**, 064017 (2015).
- [61] E. Zerah-Harush and Y. Dubi, Universal origin for environment-assisted quantum transport in exciton transfer networks, *J. Phys. Chem. Lett.* **9**, 1689 (2018).
- [62] <https://keras.io/>
- [63] M. Frisch, G. Trucks, H. Schlegel, G. Scuseria, M. Robb, J. Cheeseman, G. Scalmani, V. Barone, G. Petersson, H. Nakatsuji, *et al.*, GAUSSIAN 16, revision a. 03, Vol. 3 (Gaussian, Inc., Wallingford, Connecticut, 2016).
- [64] A. V. Brethome, S. P. Fletcher, and R. S. Paton, Conformational effects on physical-organic descriptors: The case of sterimol steric parameters, *ACS Catal.* **9**, 2313 (2019).