

Wall, Michael L., Giuseppe D'Aguanno. "Tree tensor network classifiers for machine learning: from quantum-inspired to quantum-assisted." Phys. Rev. A 104 (7 October 2021).
doi:<https://doi.org/10.1103/PhysRevA.104.042408>.

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it's important to you. Thank you.

Tree-tensor-network classifiers for machine learning: From quantum inspired to quantum assistedMichael L. Wall^{*} and Giuseppe D'Aguanno[†]*The Johns Hopkins University Applied Physics Laboratory, 11100 Johns Hopkins Road, Laurel, Maryland 20723, USA*

(Received 8 April 2021; revised 15 July 2021; accepted 3 September 2021; published 7 October 2021)

We describe a quantum-assisted machine learning method in which multivariate data are encoded into quantum states in a Hilbert space whose dimension is exponentially large in the length of the data vector. Learning in this space occurs through applying a low-depth quantum circuit with a tree-tensor-network (TTN) topology acting as an unsupervised feature extractor to identify the most relevant quantum states in a data-driven fashion and then applying a supervised linear classifier encoding the class decision in a small-dimensional quantum register. We present tools for making TTN classifiers amenable to implementation on gate-based quantum computing devices, including an embedding map with accuracy similar to the recently defined exponential machines (A. Novikov *et al.*, [arXiv:1605.03795](https://arxiv.org/abs/1605.03795)) but which produces valid quantum state embeddings of classical data vectors, and the use of manifold-based gradient optimization schemes to produce isometric operations mapping quantum states to a register of qubits defining a class decision. We detail methods for efficiently obtaining one-point and two-point correlation functions of the vectors defining the decision boundary of the quantum model, which can be used for model interpretability, as well as methods for obtaining classification decisions from partial data vectors. Further, we show that the use of isometric tensors can significantly aid in the human interpretability of the correlation functions extracted from the decision weights and may produce models that are less susceptible to adversarial perturbations. Finally, we discuss in detail the problem of compiling classically optimized isometric TTN models into unitary operations to be run on quantum computers and how isometric models requiring postselection on quantum hardware can be used to precondition variational *Ansätze* for models without postselection. We demonstrate our methodologies in applications utilizing the MNIST database of handwritten digits and a multivariate time-series data set of human activity recognition.

DOI: [10.1103/PhysRevA.104.042408](https://doi.org/10.1103/PhysRevA.104.042408)**I. INTRODUCTION**

Quantum computers, devices that utilize inherently quantum phenomena such as entanglement to process information, have long held interest due to their ability to enable disruptive exponential speedups over classical devices in certain tasks, such as integer factorization [1,2] and simulation of quantum chemical and materials science problems [3,4]. While scaling quantum computing devices to the millions of qubits required for reliable execution of these tasks [5] remains a long-term goal, we have entered an era of noisy intermediate-scale quantum (NISQ) devices [6] with tens to hundreds of qubits, noise levels too large to enable error correction, and limitations in hardware connectivity and allowed gate sets. The NISQ computing landscape has grown beyond the academic laboratory into an industrial reality [7–11], with several general-purpose research-scale machines available through cloud services and other specialized machines engineered for a specific demonstration [12]. While the ultimate goal of NISQ devices will be to mature quantum hardware technology and algorithms towards the goal of universal error-corrected quantum computing, a parallel effort to discover impactful

near-term applications of NISQ devices is the focus of significant current research [13].

Quantum-assisted machine learning (QAML), in which a quantum system forms part of a model used for statistical inferences whose fidelity is improved by interactions with data, has emerged as a promising possible avenue for NISQ devices [14–16]. There are many key reasons for this: Measurements on quantum systems have an underlying statistical interpretation, and well-performing machine learning (ML) models should be robust against noise, which may include the hardware noise present in NISQ devices. Several recent results have given theoretical support for enhancements of QAML models versus classical models on certain tasks [17–20]. In the research described herein, we focus on QAML models built upon tensor networks, an enabling technology that has transformed quantum condensed matter and many-body physics in the past 30 years [21–23]. Tensor networks offer a robust framework for defining QAML models with several additional advantages: They can be executed on classical or quantum devices with an exponentially improving expressibility on quantum devices [17], and certain tensor-network topologies enable sequential preparation schemes [24–26] that are highly quantum resource efficient [27], a key consideration for near-term devices.

Given the above favorable characteristics of tensor networks (TNs) as machine learning models, it is no surprise that a significant and growing body of work exists on TN-

^{*}Michael.Wall@jhuapl.edu[†]Giuseppe.DAguanno@jhuapl.edu

based machine learning [28–51]. While many of these works are quantum inspired [52–54] in the sense that the objects appearing in the TN description are not required to correspond to physically realizable quantum states, some proposals deal with truly quantum data structures and some have tested TN-based approaches on NISQ hardware [30,49,55]. In the present work we compare models with a tree-tensor-network (TTN) structure for classification that are constrained to be true quantum data structures with those that are unconstrained, using metrics of performance and interpretability.

In our quantum approach to TTN-based classification, we introduce several innovations. First, we identify an embedding map for translating classical data vectors into vectors in a high-dimensional Hilbert space, inspired by the classical approach of Ref. [56] that efficiently maps the data to a high-order polynomial, which provides a comparable performance while also producing valid quantum states. We also discuss means for optimizing supervised learning weights mapping a collection of quantum feature vectors in a high-dimensional space to a small register of class decision qubits. Namely, we utilize gradient descent algorithms on the Riemannian manifold of isometric tensors such that this mapping between Hilbert spaces corresponds to an allowed transformation between orthonormal quantum states. We show that these techniques, together with the quantum interpretation of the resulting model structure, enable applications such as classification of partial data vectors and interpretability analyses. Results are given for the canonical modified National Institute of Standards and Technology (MNIST) database of handwritten digits [57], in which we investigate both the classification of all digits zero through nine as well as the simpler problem of distinguishing zeros from ones, as well as a time-series data set using smartphone accelerometer and gyroscope data to recognize human activity.

A key consideration for tensor-network models for machine learning is that they are amenable to implementation on either classical or quantum hardware, with exponentially improving expressibility on the latter. Hence, in the latter part of this work we describe means of “compiling” a classically trained TTN model into unitary gates to be run on quantum hardware using as short of gate sequences as possible. Classical training of TTNs enables well-developed robust optimization strategies to be employed for preconditioning models for quantum hardware, but also produces models whose straightforward translation to quantum hardware requires postselection. We demonstrate that an isometric model compiled from a classically optimized one can be used as a variational *Ansatz* for a model without postselection, and the latter model can then be directly optimized, potentially directly on the quantum hardware.

II. TREE TENSOR NETWORKS FOR QUANTUM-ASSISTED MACHINE LEARNING

Tensor networks represent the high-rank tensor expressing the quantum wave function of a multipartite system as a contraction over low-rank tensors and hence define families of low-rank approximations whose computational power can be expressed in terms of the maximum dimension of any contracted index χ , known as the bond dimension. A wide

variety of TN topologies have been considered which are able to efficiently capture certain classes of quantum states [21–23]; in the present work we focus on tree tensor networks [58], which utilize the tree topology exemplified in Fig. 1. Here and throughout, we will utilize the Penrose graphical notation for tensors [21], in which a symbol represents a tensor, the number of lines coming off from a tensor is its rank, and when a line connects two tensors summation over that common index is implied. A symbol with lines pointing upward represents a tensor and a symbol with the lines pointing downward represents its complex conjugate.

A key feature of TTN models for interpretation is that they are representations of real-space renormalization groups. Renormalization groups mathematically formalize the changes in the description of a physical system as we change the scale at which it is observed, e.g., length scale or energy scale [59], and include a procedure for keeping the most relevant degrees of freedom when performing this coarse-graining transformation. In statistical physics it is observed that the behavior of many physical systems can be grouped into a small number of universality classes that depend on only a few parameters when viewed from very coarse scales; these universality classes may correspond to phases of matter, for example. Similarly, in machine learning we want to extract features from data at all scales in order to classify into broad categories specified by only a few relevant variables; tensor networks are a means of formalizing this approach. Based on this analogy, we will refer to the vertical layers of tensors in a TTN as levels of scale, indexed by the superscript ξ of $|\Phi^{(\xi)}(\mathbf{x})\rangle$ as in Fig. 1.

We now turn to formalizing the procedure for using a TTN for quantum-assisted machine learning of classical data. Broadly, our process for QAML has the key elements of (i) encoding of classical data into a quantum state, (ii) learning of a quantum model (tensor network) on encoded training data, and (iii) readout of quantum model results. Using these elements, we can also choose between QAML models with different hyperparameters (e.g., the bond dimension of the TN) using separate validation data and also test the ultimate performance of these models on a held-out test set. In the sections to follow, we flesh out each of the items in the workflow in more detail.

A. Mapping of classical data to quantum states

In this section we address the encoding of a classical data vector $\mathbf{x} \in \mathbb{R}^L$ into a quantum state, depicted visually by the dashed red-bordered box in Fig. 1 [60–62]. We will denote the encoded state at the lowest level of scale by $|\Phi^{(0)}(\mathbf{x})\rangle$. We will consider that each element x_j of the data vector is encoded into a d -dimensional quantum state ($d = 2$ for a qubit) such that the full wave function $|\Phi^{(0)}(\mathbf{x})\rangle$ exists in a d^L -dimensional Hilbert space. The only restriction we will place on the encoding of classical data in quantum states is that each classical data vector is encoded in an unentangled product state, denoted by the absence of horizontal connecting lines in Fig. 1. Utilizing unentangled states has several advantages: They are the simplest to prepare experimentally with high fidelity and also enable us to use qubit-efficient sequential preparation schemes [24–27]. Encoding individual data vectors in product

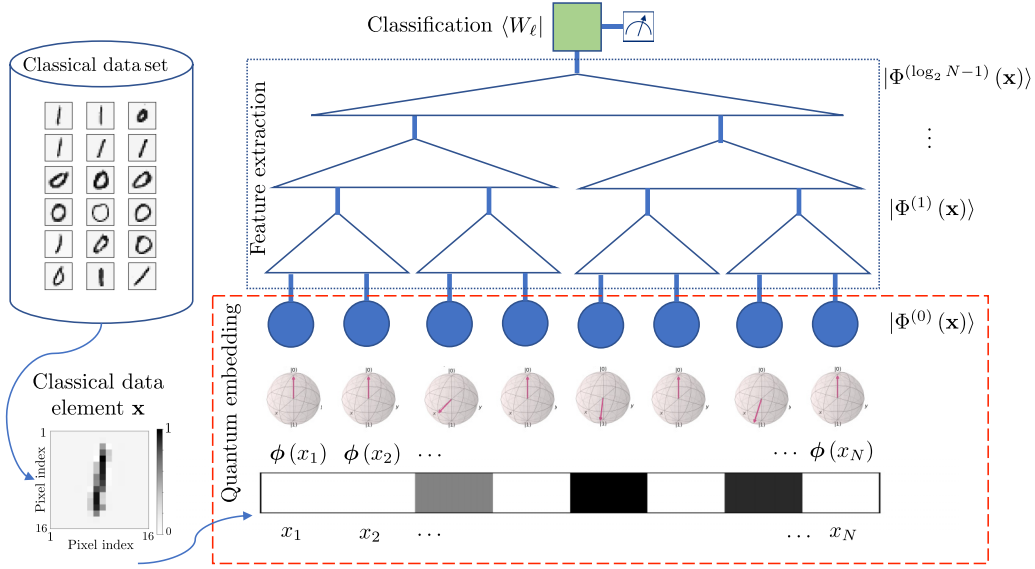


FIG. 1. Overview of the machine learning workflow for a TTN classifier. A data instance \mathbf{x} from a classical data set is transformed into a quantum state $|\Phi^{(0)}(\mathbf{x})\rangle$ by feeding each element of the data vector x_i through a local map $\phi(x_i)$ defining a qubit superposition, as shown schematically on the Bloch sphere. The isometric tensors of the TTN (triangles) define a coarse graining of collections of these quantum feature vectors at progressively higher levels of scale. At the highest level of scale, the projection of the extracted feature vector onto a collection of weight vectors $\langle W_\ell|$ defines a classification decision.

states also ensures that any entanglement in the quantum ML model arises from correlations in an ensemble of data and not from *a priori* assumptions about preexisting correlations for individual data vectors [28]. We will parametrize the map from an L -dimensional classical data vector \mathbf{x} to an ensemble of L two-level systems (qubits) as

$$|\Phi^{(0)}(\mathbf{x})\rangle = \bigotimes_{j=1}^L \left(\sum_{i_j=1}^2 \phi_{i_j}^{(j)}(x_j) |i_j\rangle \right). \quad (1)$$

That is, the parametrization is accomplished in terms of local maps $\phi^{(j)}(x)$ mapping a single data element into a superposition of qubit states (see Fig. 1). In order to ensure that the full map $\Phi^{(0)}(\mathbf{x})$ maps each data instance into a normalized vector in Hilbert space, we require that

$$\sum_i |\phi_i^{(j)}(x)|^2 = 1 \quad \forall x. \quad (2)$$

This condition is satisfied by the phaselike encoding

$$\begin{aligned} \phi_0(x) &= \cos\left(\frac{\pi}{2} \frac{x - x_{\min}}{x_{\max} - x_{\min}}\right), \\ \phi_1(x) &= \sin\left(\frac{\pi}{2} \frac{x - x_{\min}}{x_{\max} - x_{\min}}\right) \end{aligned} \quad (3)$$

that has been used in Refs. [28,36,46,63] to encode data for quantum-inspired ML applications and is shown schematically by the Bloch sphere representations of each classical data element in Fig. 1. In Ref. [56] a quantum-inspired algorithm using MPSs (known in the numerical analysis community as tensor trains [54]) found good performance in certain learning tasks using the map

$$\phi_0(x) = 1, \quad \phi_1(x) = ax. \quad (4)$$

This has the appealing property that each data vector \mathbf{x} is mapped into a weighted superposition of all correlations $x_i x_j \cdots x_k$ of orders 0 through L with each x_i appearing at most once, and so a tensor-network weighting vector in this space can efficiently select out the most relevant correlations from this exponentially large set. The factor a can be used to scale the data and avoid numerical overflow. Unfortunately, this map does not satisfy the condition in Eq. (2) and so is not suitable for application on quantum hardware. We can define a generalization of Eq. (3) as

$$\phi_0(x) = \cos\left(a \frac{x}{x_{\max}}\right), \quad \phi_1(x) = \sin\left(a \frac{x}{x_{\max}}\right), \quad (5)$$

with $a \sim O(0.1)$, which satisfies Eq. (2) and so is suitable for quantum discriminative applications. Noting that $a \frac{x}{x_{\max}} \ll 1 \quad \forall x$, we can use the small-angle approximation to find

$$\phi_0(x) = 1 + O(a^2), \quad \phi_1(x) = a \frac{x}{x_{\max}} + O(a^3), \quad (6)$$

and so this map has the same essential features as Eq. (4). A comparison of model performance utilizing the maps (5) and (4) will be given in Sec. III.

B. Construction of unsupervised feature extractor

We now turn to building an unsupervised feature extractor as a TTN acting on the encoded quantum data description, following Ref. [29]. This procedure is depicted graphically by the dotted blue box in Fig. 1. We assume that we have a collection of M training data vectors $\{\mathbf{x}_m\}$, $m = 1, \dots, M$, that has been encoded into a collection of training states $\{|\Phi(\mathbf{x}_m)\rangle\}$. Using this map and these states, we would like to build a model $f(\mathbf{x}) = \langle W | \Phi(\mathbf{x}) \rangle$ parametrized by a weight vector $|W\rangle$ in the many-body Hilbert space, to perform some machine learning task (e.g., classification). Given our collection of training vectors, the optimal weights for a given task can be

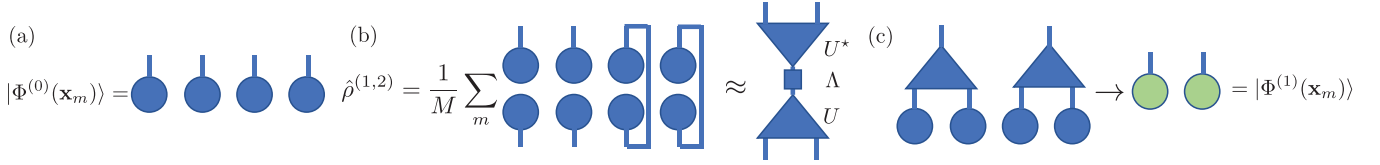


FIG. 2. Tensor-network diagrams for (a) mapping of a data vector to a product state in Hilbert space, (b) approximation of a two-site reduced density operator in terms of an isometric tensor, and (c) renormalization of a data vector using the TTN isometries, resulting in a new product state in the renormalized Hilbert space.

represented by a linear combination of the training vectors in the quantum space as $\langle W| = \sum_m \alpha_m \langle \Phi(\mathbf{x}_m)|$, where the $\{\alpha_m\}$ are task specific. This is the representer theorem, which states that the weights lie within the span of feature vectors generated by the training data. If the effective dimension of the set of relevant feature vectors is much smaller than the full dimension of the space, this will enable us to efficiently compress our model representation. Ignoring issues of efficiency, one way to find the effective dimension of this set would be from the singular value decomposition (SVD) of the matrix $\Phi_{im} \equiv \langle i_1 \cdots i_L | \Phi^{(0)}(\mathbf{x}_m) \rangle$, where $\mathbf{i} = (i_1, \dots, i_L)$ is a multi-index spanning the full d^L -dimensional Hilbert space. Namely, given $\Phi_{im} \rightarrow \sum_\mu U_{i\mu} S_\mu V_{\mu m}$, we have $W_i = \sum_\mu \beta_\mu U_{i\mu}$. The matrix Φ_{im} is $d^L \times M$ dimensional, and so for typical training data-set sizes $M \gtrsim 10^4$ a direct decomposition of this matrix is unfeasible.

Instead of the direct approach using the SVD, we will instead utilize an iterative approach that builds a TTN to extract the relevant feature vectors. We start by noting that the SVD matrix U , whose columns form a basis for the statistically significant set of feature vectors from the training data set, can alternately be obtained as the eigenvectors of the feature space covariance matrix

$$\rho_{ii'} = \frac{1}{M} \sum_{m=1}^M \Phi_{im} \Phi_{i'm} = \sum_{\mu} U_{i\mu} S_{\mu}^2 U_{i'\mu}^*. \quad (7)$$

Translating back to the language of quantum mechanics, we see that this object is nothing but the density operator obtained

from an incoherent sum of all training vectors

$$\hat{\rho} = \frac{1}{M} \sum_m |\Phi(\mathbf{x}_m)\rangle \langle \Phi(\mathbf{x}_m)|. \quad (8)$$

Recall that each of our feature vectors $|\Phi^{(0)}(\mathbf{x}_m)\rangle$ at the lowest level of scale is unentangled according to the restrictions placed in Sec. II A; however, the sum over feature vectors introduces correlations that may manifest themselves as entanglement in feature space. If we look at only a subset of the data elements in the feature space, correlations with other elements will induce fluctuations in this subset that are captured as impurity in the corresponding reduced density operator. The tree-tensor-network construction will capture these correlations in feature space hierarchically, using a procedure motivated by how the dominant fluctuations manifest themselves in certain quantum mechanical systems [29].

We begin the TTN construction procedure by defining a set of operations which group pairs of data elements together. We will construct these operations to capture as much of the subsystem fluctuations as possible given restricted resources, which will be defined explicitly below. We do so by constructing the reduced density operators of pairs of neighboring sites¹

¹Note that while we index the data elements using $j = 1, \dots, L$, this is simply a numbering scheme and does not require a one-dimensional topology for the data space.

$$\begin{aligned} \hat{\rho}^{(2j-1,2j)} &= \frac{1}{M} \sum_{m=1}^M \sum_{i_{2j-1} i'_{2j-1} i_{2j} i'_{2j}} \Phi_{i_{2j-1}}^{(0)}(x_{m,2j-1}) \Phi_{i'_{2j-1}}^{(0)}(x_{m,2j-1}) \Phi_{i_{2j}}^{(0)}(x_{m,2j}) \Phi_{i'_{2j}}^{(0)}(x_{m,2j}) \\ &\quad \times |i_{2j-1} i_{2j}\rangle \langle i'_{2j-1} i'_{2j}| \prod_{j' \neq 2j-1, 2j} \left(\sum_{i=1}^2 |\Phi_i^{(0)}(x_{m,j'})|^2 \right). \end{aligned} \quad (9)$$

This represents the effective covariance of the $(2j-1)$ and $(2j)$ elements of the data vector in feature space, taking into account the average effect of correlations with all other elements of the data vector. Just as with the full density operator (8), these reduced density operators are Hermitian and so admit an eigendecomposition

$$\hat{\rho}_{(i_{2j-1} i_{2j})(i'_{2j-1} i'_{2j})}^{(2j-1,2j)} = \sum_{\mu_j} U_{(i_{2j-1} i_{2j})\mu_j}^{(0)[j]\star} \Lambda_{\mu_j} U_{(i'_{2j-1} i'_{2j})\mu_j}^{(0)[j]}, \quad (10)$$

denoted graphically in Fig. 2(b). The eigenvectors, specified as columns of the matrices $\mathbb{U}^{(0)[j]}$, define a change of basis

from the bare feature space $|i_{2j-1} i_{2j}\rangle$ to a renormalized feature space $|\mu_j\rangle$ as

$$|\mu_j\rangle = \sum_{i_{2j-1} i_{2j}} U_{(i_{2j-1} i_{2j})\mu_j}^{(0)[j]} |i_{2j-1} i_{2j}\rangle. \quad (11)$$

In general, $\hat{\rho}^{(2j-1,2j)}$ has d^2 nonzero eigenvalues and so a complete change of basis will require all of the eigenvectors to fully reproduce the training data. However, in many physically relevant cases, not all possible feature vectors are seen or some combinations occur with negligibly low probability. In

this case, we can keep only a restricted number χ of the eigenvectors corresponding to the largest eigenvalues and hence define an isometric transformation from the original d^2 -dimensional space to a χ -dimensional space. Alternatively, we can obtain χ implicitly by requiring that we discard at most ε of the Frobenius norm of the reduced density operator through

$$1 - \left(\sum_{\alpha=1}^{\chi} \rho_{\alpha\alpha} \right) / \left(\sum_{\alpha} \rho_{\alpha\alpha} \right) \leq \varepsilon. \quad (12)$$

The strategy of using the states corresponding to the largest eigenvalues of a reduced density operator to renormalize the space is known as White's rule and is the underpinning of the celebrated density matrix renormalization group method in quantum physics [21,64].

Collecting all of the isometric tensors defined from applying the procedure above to a complete disjoint set of data element pairs, we have defined an isometric transformation to a globally renormalized feature space

$$\langle \mu | \Phi^{(1)}(\mathbf{x}) \rangle = \sum_{\mathbf{i}} \left(\prod_{j=1}^{L/2} U_{(i_{2j-1}, i_{2j}) \mu_j}^{(0)[j]} \right) \langle \mathbf{i} | \Phi^{(0)}(\mathbf{x}) \rangle, \quad (13)$$

as shown graphically in Fig. 2(c). Importantly, the renormalized feature space still has a product structure between its component vectors, even though the components are now of higher dimension (χ instead of d) and account for correlations between subcomponents at a lower scale. Hence, we can iterate this renormalization procedure up until the highest scale. At the highest level of scale, we are left with a single index μ labeling the relevant set of feature vectors for the data set. The entire hierarchical network, when all intermediate bond indices have been contracted, defines an isometry approximately corresponding to the eigendecomposition of the full density operator (8). The network does not have to be contracted up to the highest level of scale to be useful; stopping before the highest level of scale maps the original high-dimensional space into another multipartite quantum state representation. References [29,44] have investigated models in which a TN with a treelike structure is augmented by another TN structure, such as a matrix product state. Finally, we note that this procedure defines a family of models indexed by the maximum bond dimensions χ of the isometries [alternatively, through the norm cutoff ε defined in Eq. (12)] which form the main hyperparameter of the models. This cutoff can be used as a regularization to avoid overfitting.

C. Training of a supervised classifier

So far in the QAML workflow we have encoded our classical data instances into quantum states in a d^L -dimensional quantum Hilbert space and then utilized a tree-tensor-network feature extractor that acts as an isometry projecting these quantum feature vectors onto a small relevant set. The isometric feature extractor was constructed in an unsupervised manner, utilizing the features extracted from the training data and White's rule for determining which features were the most relevant. Now that we have extracted the relevant features, we turn to optimizing a mapping from the extracted features to a classification decision, denoted by the green box at the

top of the network in Fig. 1. Let us again consider that we have training data $\{\mathbf{x}_m\}$, $m = 1, \dots, M$, that are in one of $\ell = 1, \dots, C$ classes, with the truth class label of data vector m denoted by ℓ_m . If we denoted the extracted feature vector at the highest level of scale by $|\Phi(\mathbf{x})\rangle$, we now wish to define weights $\langle W_\ell |$ such that the functions $f_\ell(\mathbf{x}) = \langle W_\ell | \Phi(\mathbf{x}) \rangle$ are 1 when \mathbf{x} is in class ℓ and 0 otherwise. For simplicity, we do so by optimizing an unregularized quadratic cost function [29]

$$\begin{aligned} \mathcal{C} &= \frac{1}{2M} \sum_{m=1}^M \sum_{\ell=1}^C [f_\ell(\mathbf{x}_m) - \delta_{\ell\ell_m}]^2 \\ &= \frac{1}{2M} \sum_{m=1}^M \sum_{\ell=1}^C [\langle W_\ell | \Phi(\mathbf{x}_m) \rangle - \delta_{\ell\ell_m}]^2. \end{aligned} \quad (14)$$

The least-squares solution that optimizes each weight vector independently is

$$W_{v,\ell} = [\langle v | \Phi(\mathbf{x}_m) \rangle]^{-1} \delta_{\ell\ell_m}, \quad (15)$$

in which $[\cdot]^{-1}$ denotes the Moore-Penrose pseudoinverse and $\delta_{\ell\ell_m}$ is a length M vector that is 1 when $\ell_m = \ell$ and 0 otherwise. With these optimized weight vectors in hand, we can classify a test vector \mathbf{x}_{test} according to $\arg \max_{\ell} |\langle W_\ell | \Phi(\mathbf{x}_{\text{test}}) \rangle|^2$. As above, if the renormalization procedure is stopped before the highest level of scale, then the weight vector $\langle W_\ell |$ is a multipartite tensor mapping the degrees of freedom remaining after renormalization into a class decision label ℓ .

While the above pseudoinverse solution is optimal from the perspective of the cost function, the resulting matrix \mathbb{W} is not isometric and so the set of vectors W_ℓ does not constitute orthonormal quantum states in the coarse-grained feature space. We remedy this here by considering an optimization procedure that seeks to minimize the cost function (14) directly within the space of isometric matrices. Complex isometric matrices of dimension $\chi \times C$ (with $\chi \geq C$) define a Riemannian manifold known as the Stiefel manifold [65]

$$\text{St}(\chi, C) = \{\mathbb{W} \in \mathbb{C}^{\chi \times C} : \mathbb{W}^\dagger \mathbb{W} = \mathbb{I}\}. \quad (16)$$

Several recent works [66,67] have discussed the application of manifold optimization techniques to isometric tensors for quantum applications, which we briefly review here to keep the exposition self-contained. First, we will define a few tools from Riemannian geometry that will be useful, the first being the orthogonal projection of a general complex matrix $\mathbb{D} \in \mathbb{C}^{\chi \times C}$ onto the tangent space $\mathcal{T}_{\mathbb{W}}\mathcal{M}$ of the manifold $\mathcal{M} = \text{St}(\chi, C)$ at the point \mathbb{W} . For the Stiefel manifold, this projection takes the form

$$\mathcal{P}_{\mathbb{W}}(\mathbb{D}) = \mathbb{D} - \frac{1}{2} \mathbb{W} (\mathbb{W}^\dagger \mathbb{D} + \mathbb{D}^\dagger \mathbb{W}). \quad (17)$$

The next operation we will need is retraction, which is a smooth map $\mathcal{R}_{\mathbb{W}}(\cdot)$ from $\mathcal{T}_{\mathbb{W}}\mathcal{M} \rightarrow \mathcal{M}$ that offers a computationally efficient alternative to the exponential map generalizing point transformations in Euclidean space. Retractions are not unique, but can be defined by any smooth map satisfying the conditions

$$\mathcal{R}_{\mathbb{W}}(\mathbf{0}) = \mathbb{W} \quad \forall \mathbb{W} \in \mathcal{M}, \quad (18)$$

$$\frac{d}{dt} \mathcal{R}_{\mathbb{W}}(t\mathbf{v})|_{t=0} = \mathbf{v} \quad \forall \mathbf{v} \in \mathcal{T}_{\mathbb{W}}\mathcal{M}, \quad (19)$$

in which $\mathbf{0}$ denotes the zero vector in $\mathcal{T}_{\mathbb{W}}\mathcal{M}$. In practice, we utilize the SVD retraction defined by

$$\mathcal{R}_{\mathbb{W}}(\boldsymbol{\eta}) = \mathbf{U}\mathbf{V}, \quad \mathbb{W} + \boldsymbol{\eta} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}, \quad (20)$$

where the rightmost equality defines the SVD. The final definition we need is vector transport, which is a computationally efficient alternative to parallel transport which generalizes transport of a point x along a vector direction v from Euclidean geometry. As with retraction, vector transport is not unique, but is any smooth map $\tau_{\mathbb{W}}(\mathbf{y}, \boldsymbol{\eta})$ ($\mathbb{W} \in \mathcal{M}$ and $\mathbf{y}, \boldsymbol{\eta} \in \mathcal{T}_{\mathbb{W}}\mathcal{M}$) with the properties

$$\tau_{\mathbb{W}}(\mathbf{y}, \boldsymbol{\eta}) \in \mathcal{T}_{\mathcal{R}_{\mathbb{W}}(\boldsymbol{\eta})}\mathcal{M}, \quad (21)$$

$$\tau_{\mathbb{W}}(\mathbf{y}, \mathbf{0}) = \mathbf{y}, \quad (22)$$

$$\tau_{\mathbb{W}}(a\mathbf{y} + b\mathbf{z}, \boldsymbol{\eta}) = a\tau_{\mathbb{W}}(\mathbf{y}, \boldsymbol{\eta}) + b\tau_{\mathbb{W}}(\mathbf{z}, \boldsymbol{\eta}). \quad (23)$$

In the results given below, we use the vector transport

$$\tau_{\mathbb{W}}(\mathbf{y}, \boldsymbol{\eta}) = \mathcal{P}_{\mathcal{R}_{\mathbb{W}}(\boldsymbol{\eta})}(\mathbf{y}), \quad (24)$$

with the SVD retraction defined above.

With the above definitions in hand, we can now define a gradient optimization procedure directly within the space of isometric matrices. We recall the standard iterative scheme for gradient descent of a cost function $\mathcal{C}(\mathbf{w})$ with momentum parameter β and learning rate η :

$$\mathbf{m}_{i+1} = \beta\mathbf{m} + (1 - \beta)\nabla\mathcal{C}(\mathbf{w}_i), \quad (25)$$

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \eta\mathbf{m}_{i+1}. \quad (26)$$

Here i is the iteration index. We can generalize this to manifold gradient descent as

$$\tilde{\mathbf{m}}_{i+1} = \beta\mathbf{m}_i + (1 - \beta)\mathcal{P}_{\mathbb{W}_i}[\nabla\mathcal{C}(\mathbb{W}_i)], \quad (27)$$

$$\mathbb{W}_{i+1} = \mathcal{R}_{\mathbb{W}_i}(-\eta\tilde{\mathbf{m}}_{i+1}), \quad (28)$$

$$\mathbf{m}_{i+1} = \tau_{\mathbb{W}_i}[\tilde{\mathbf{m}}_{i+1}, -\eta\tilde{\mathbf{m}}_{i+1} - \eta\beta(\tilde{\mathbf{m}}_{i+1} - \mathbf{m}_i)]. \quad (29)$$

For the quadratic cost function defined in Eq. (14), we have that

$$\begin{aligned} [\nabla\mathcal{C}]_{v\ell} &= 2\frac{\partial\mathcal{C}}{\partial W_{v\ell}^*} \\ &= \frac{1}{M}\sum_{m=1}^M \langle v|\Phi(\mathbf{x}_m)\rangle[\langle\Phi(\mathbf{x}_m)|W_\ell\rangle - \delta_{\ell\ell_m}]. \end{aligned} \quad (30)$$

A convenient starting point to initialize the isometric optimization procedure is the nearest isometric matrix to the pseudoinverse solution \mathbb{W}_{pinv} in the L_2 -norm, given by

$$\mathbb{W}_{\text{iso}} = \mathbf{U}\mathbf{V}, \quad \mathbb{W}_{\text{pinv}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}. \quad (32)$$

Example cost function behavior during optimization will be given later in Sec. III.

D. Application of classifier on test data

At this stage we have a complete pipeline to go from a classical data instance to a class decision: The data instance \mathbf{x} is embedded into a product state feature vector using the

map $\phi(x_j)$ resulting in the state $|\Phi^{(0)}(\mathbf{x})\rangle$; this state is coarse grained using the TTN feature extractor into the state $|\Phi^{(\xi)}\rangle$ at the level of scale ξ and then the inner product of this coarse-grained feature vector with the weights \mathbb{W} defines a decision function $\arg\max_{\ell} |\langle W_\ell|\Phi(\mathbf{x}_{\text{test}})\rangle|^2$ mapping to a class label ℓ . If the test data have the same dimension as the training data (i.e., there are no missing elements in the data vector), the tensor network used to evaluate the decision function amounts to the recursive renormalization of feature vectors followed by an inner product, as shown schematically in Figs. 3(a)–(d). In this section we broaden the applicability of this procedure to include cases in which data are missing for certain elements of the test vector. Given that we are dealing with quantum data structures, a lack of subsystem information in our approach is naturally handled through the formalism of density operators. However, the renormalization procedure for density operators differs from that of feature vectors and will be explained herein.

For the case in which our test vector contains partial information, the tensor network to be contracted for the decision functions takes a form as in Figs. 3(e)–(h), in which the dimensions with missing data are contracted. We recall that the tensors $U_{(ii')\mu_j}^{(\xi)[j]}$ are isometric and so obey the row-orthogonality condition $\sum_{ii'} U_{(ii')\mu_j}^{(\xi)[j]} U_{(ii')\mu'_j}^{(\xi)[j]*} = \delta_{\mu_j\mu'_j}$. Hence, if no data are present in any dimension of a tensor at level ξ , the trace is simply shifted to the next level of scale, as seen in going from Fig. 3(e) to Fig. 3(f). If one of the elements of U , say, the element corresponding to index i , contains a feature vector while the other element i' contains no information, the result is not another feature vector but instead a density operator, denoted by a square in Fig. 3. Renormalization of any object together with a density operator produces another density operator. This calculus of renormalization can be summarized with a collection of rules $(\cdot, \cdot) \rightarrow \mathcal{O}$, in which (\cdot, \cdot) denotes the two objects input to an isometric tensor and \mathcal{O} denotes its output, namely, the rules are

$$\begin{aligned} (|, |) &\rightarrow |, & (\circ, \circ) &\rightarrow \circ, & (|, \circ) &\rightarrow \square, \\ (\circ, |) &\rightarrow \square, & (\square, *) &\rightarrow \square, & (*, \square) &\rightarrow \square, \end{aligned} \quad (33)$$

in which $|$ denotes a contracted index (no data), \circ denotes a feature vector, \square denotes a density operator, and $*$ denotes any object. From an implementation standpoint, the order in which indices are contracted is essential for obtaining the best scaling algorithm [21]. In particular, it is advisable to sum over at most one contracted index at a time, storing the results in intermediate tensors, in order to reduce the total number of operations. Further, density operators obtained as intermediaries will not generally be of full rank and so can be eigendecomposed and the resulting set of feature vectors renormalized and reconstructed more efficiently than direct renormalization of the density operator.

As seen in Figs. 3(e)–(h), when utilizing partial information to extract a class decision, the relevant quantity can be written as $\arg\max_{\ell} \langle W_\ell|\hat{\rho}(\mathbf{x}_{\text{test}})|W_\ell\rangle$, in which $\hat{\rho}(\mathbf{x}_{\text{test}})$ results from the renormalization of $\text{Tr}_{\mathcal{S}}|\Phi^{(0)}(\mathbf{x}_{\text{test}})\rangle\langle\Phi^{(0)}(\mathbf{x}_{\text{test}})|$. Here \mathcal{S} is the set of all indices where \mathbf{x}_{test} does not have support. We find that when a significant portion of the data is absent, the resulting density matrix $\hat{\rho}$ is highly mixed. Hence,

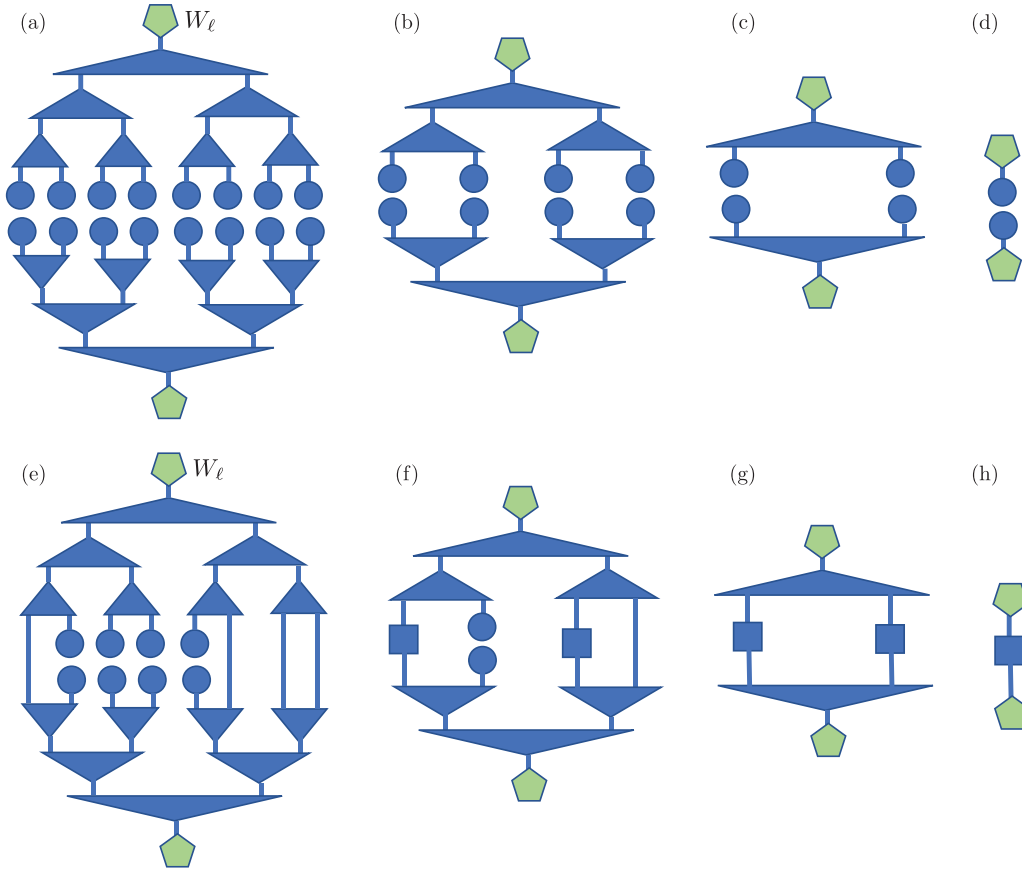


FIG. 3. Classification decision function evaluation with complete and partial data. (a)–(d) Evaluation of a TTN model with a full data vector amounts to renormalization of feature vectors followed by an inner product. (e)–(h) Evaluation of a TTN model with partial information requires renormalization not only of feature vectors, but also of density operators (squares).

the class decision function will be a sum over many terms $\arg \max_{\ell} [\sum_{\lambda} p_{\lambda} |\langle W_{\ell} | \lambda \rangle|^2]$, in which p_{λ} are the eigenvalues of $\hat{\rho}$ and $|\lambda\rangle$ the associated eigenvectors. We have found that the best performance is observed when we instead project the weight vectors onto only the eigenvector with highest probability, i.e., $\arg \max_{\ell} [p_1 |\langle W_{\ell} | 1 \rangle|^2]$, at least for the present training strategy which utilizes only the full data vectors. More details on performance will be presented in Sec. III.

E. Interpretability analysis

In this section, we highlight a property of the TTN feature extractor which can aid in interpretability of the classification decision. Namely, since the TTN is formed through isometric tensors, it is also possible to read the tensor network in reverse and so “fine grain” a decision vector from the highest level of scale back to the lowest, “data” level of scale. This is likely a more general feature of quantum machine learning algorithms, as the unitary nature of the operations involved means that the computation should be reversible, taking as input a classification decision vector and producing a vector at the feature scale. In principle, this fine graining proceeds as in Figs. 4(a)–(d). However, the resulting vector exists in the full d^L -dimensional Hilbert space and so cannot be immediately interpreted. A natural way of analyzing this state is to look at low-order correlation functions or moments, analogous to

looking at the magnetization or magnetic susceptibility of a many-body quantum spin system [68]. In Figs. 4(e)–(i) we see that tracing over all data vector elements but one produces a one-point reduced density matrix $\hat{\rho}_{i i'}^{(1)}$ that can be efficiently obtained using the tensor-network structure of the feature extractor. From the one-point reduced density matrix $\hat{\rho}_{i i'}^{(1)}$, we can interpret its eigenvectors in terms of the embedding map $\phi(x)$ as in Eq. (1) and so “undo” the original feature space embedding back into the classical data space. As an example, for the scaled phase map (5), we find the average data element from the eigendecomposition $\hat{\rho}^{(1)} \rightarrow \{p_{\lambda}, |\lambda\rangle\}$ as

$$\frac{\langle x \rangle}{x_{\max}} = \sum_{\lambda} p_{\lambda} \tan^{-1} \left(\frac{\langle 1 | \lambda \rangle}{\langle 0 | \lambda \rangle} \right) / a. \quad (34)$$

Interpretations of the classification weight vectors can also be extended to higher-order correlations [38]. As an example, we can construct the two-point density matrix $\hat{\rho}_{(i i'), (j j')}^{(2)}$ using diagrams analogous to Figs. 4(e)–(i). In order to isolate the contributions from correlations that do not appear in the lowest-order moment, it is useful to define the correlation density matrix [69,70] as

$$\hat{\rho}_{(i i'), (j j')}^{(C)} = \hat{\rho}_{(i i'), (j j')}^{(2)} - \hat{\rho}_{i i'}^{(1)} \hat{\rho}_{j j'}^{(1)}. \quad (35)$$

From a quantum perspective, the correlation density matrix is useful because its trace with any product operator provides the

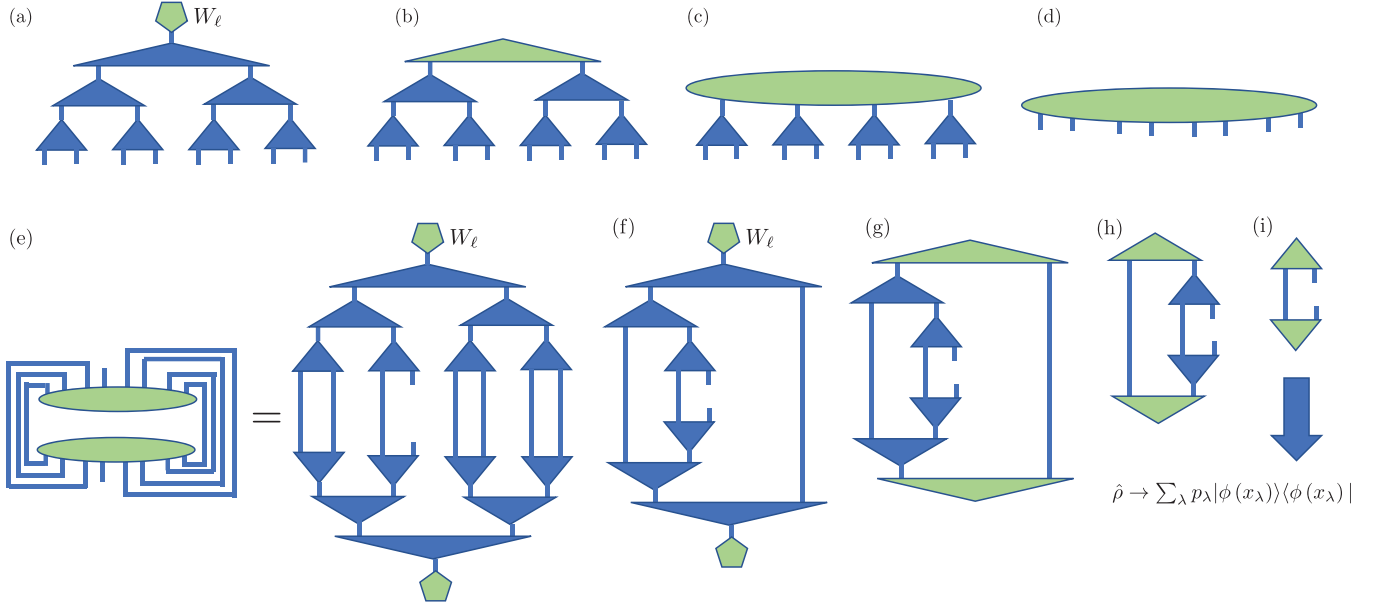


FIG. 4. Fine graining of feature vectors from higher to lower levels of scale. (a)–(d) Fine graining of a classification decision vector to a many-body state at the raw data scale. (e)–(i) Looking at a single data-scale element of the fine-grained decision vector can be performed efficiently using the TTN structure.

associated connected correlation function

$$\text{Tr}(\hat{\rho}^{(C)} \hat{O}_i \otimes \hat{O}_j) = \langle \hat{O}_i \hat{O}_j \rangle - \langle \hat{O}_i \rangle \langle \hat{O}_j \rangle. \quad (36)$$

Matrix decompositions of the correlation density matrix can be used to identify the dominant correlations present in a quantum system for detailed analysis, as discussed in Refs. [69,70]. A related scalar metric of the degree of correlation between sites i and j is given by the mutual information [71]

$$I(i, j) = H(\hat{\rho}_i^{(1)}) + H(\hat{\rho}_j^{(1)}) - H(\hat{\rho}_{ij}^{(2)}), \quad (37)$$

in which $H(\hat{\rho}) = -\text{Tr}(\hat{\rho} \ln \hat{\rho})$ is the von Neumann entropy. Examples will be provided along with the applications in Sec. III.

III. EXAMPLE APPLICATIONS

In this section we detail applications of the above approach to two data sets. The first is the canonical MNIST database of handwritten digits [57], which is commonly used as a machine learning benchmark. This data set consists of 28×28 pixel grayscale arrays of the digits 0 through 9, and we consider both classification of digits into all ten classes and the simpler problem of identifying the digits 0 and 1, for the purposes of illustration. In order to show the utility of TTN-based machine learning beyond image data, we also apply our methodologies to a multivariate human activity recognition time-series data set [72]. Here data extracted from accelerometers and gyroscopes attached to test subjects are used to infer the type of activity being performed, e.g., walking or sitting still. Our specific data preprocessing steps will be presented together with the classifier analysis in the following sections.

A. MNIST database of handwritten digits

Our first steps in utilizing the MNIST data set are to resize and scale the data uniformly. We utilize a bivariate spline to interpolate the data from its original 28×28 size into 16×16 arrays for the convenience of having the total number of pixels be a power of 2 and then scale the data such that each pixel in the training data set is between 0 and 1. Examples of data elements for the digits 0 and 1 processed in this fashion are shown in Fig. 1. The topology of the TTN is such that rows of the data are renormalized in lower layers and columns in higher layers, as exemplified in Fig. 5 using an 8×8 interpolation for clarity. The first task that we consider using this data set is building an unsupervised feature extractor on all 60 000 elements of the training data set and then defining a supervised classification into the ten classes of digit values

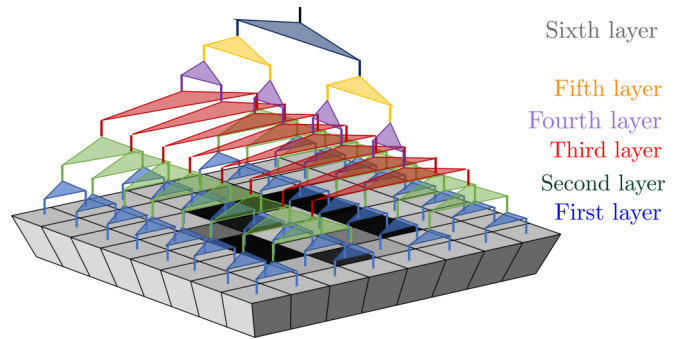


FIG. 5. Topology of TTN renormalization for images. Our TTN feature extractor first renormalizes pixels neighboring within rows (first three layers), followed by renormalization of neighboring columns (last three layers). For clarity, the procedure is shown for an image interpolated to 8×8 rather than the 16×16 used for classification.

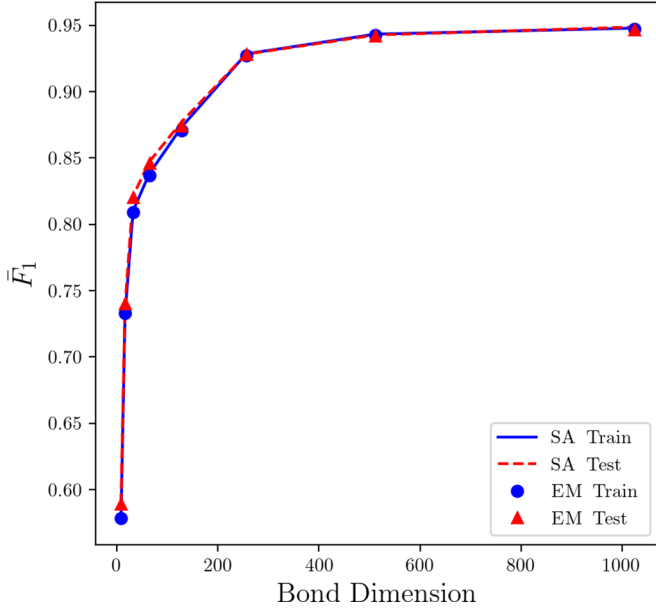


FIG. 6. Dependence of average F_1 score on tensor-network bond dimension. The average F_1 score on the training (test) data set for all ten digits is shown vs the bond dimension of the top layer with the blue solid (red dashed) line. Lines correspond to the small-angle (SA) map (5) and symbols to the exponential machine (EM) map (4), both with $a = 0.1$.

0–9, tested on all 10 000 elements of the test set. As a scalar metric of performance, we use the class-averaged F_1 score, which is obtained from precision p and recall r as

$$p_i = \frac{C_{ii}}{\sum_j C_{ji}}, \quad (38)$$

$$r_i = \frac{C_{ii}}{\sum_j C_{ij}}, \quad (39)$$

$$[F_1]_i = 2 \frac{p_i r_i}{p_i + r_i}. \quad (40)$$

Here \mathbb{C} is the confusion matrix indexed by classes whose element C_{ij} represents the number of data elements predicted to be in class j whose truth class is i .

We begin by comparing the average F_1 scores computed for the training and test sets as a function of the bond dimension of the top layer, χ , in Fig. 6. Here the blue solid (red dashed) lines correspond to the training (test) set average F_1 score using the small-angle map (5) with $a = 0.1$. The results for the exponential machine map [54] (4) with $a = 0.1$ are shown with symbols. We note that the unsupervised feature extraction layers were trained using a density matrix eigenvalue cutoff $\varepsilon = 7 \times 10^{-5}$ as defined in Eq. (12) and then only the top layer was restricted to a maximum bond dimension of χ . For small bond dimension $\chi < 128$ we obtain the optimal weights by direct construction of the pseudoinverse, while for larger bond dimensions we obtain the weights using a sparse least-squares solver [73]. The results of Fig. 6 demonstrate that the small-angle and exponential machine maps have the same qualitative features from a learning perspective and

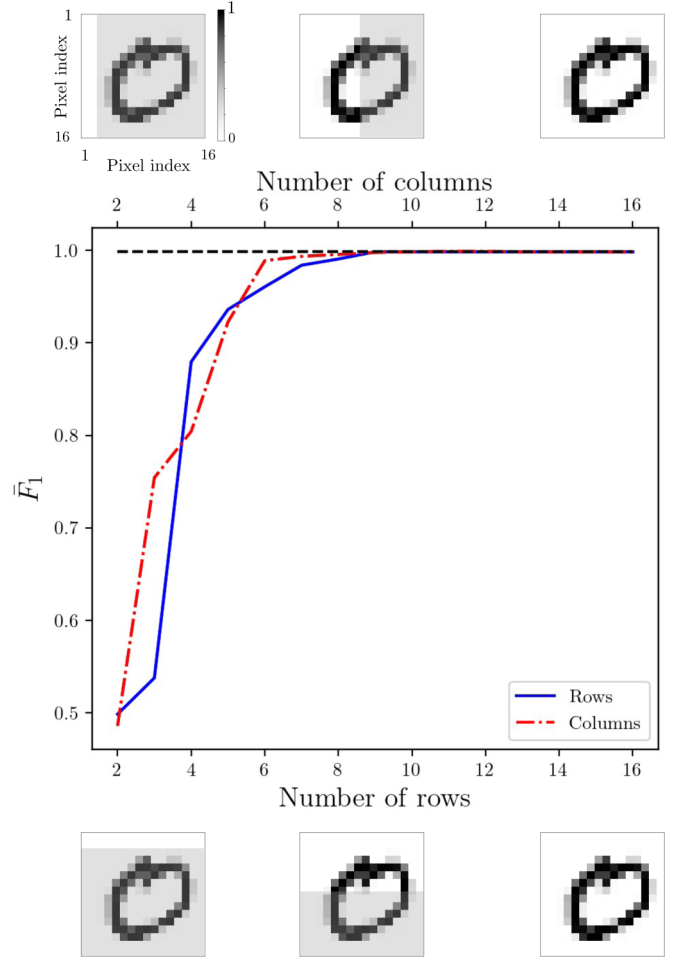


FIG. 7. Dependence of average test F_1 score on amount of data. The average F_1 score on the test data set of determining zeros vs ones is shown for a range of partial data. The bottom axis corresponds to the blue solid line using a subset of the rows of an image and the top axis corresponds to the red dot-dashed line using a subset of columns, with the ordering as indicated by shading of the test data-set elements near the axes. The black dashed line corresponds to all data being used.

show that TTNs have sufficient expressive power to classify this data set with high accuracy.

In order to enable a more detailed analysis in what follows, we now specialize to a simpler classification task of distinguishing the digits 0 and 1 from the MNIST data set. We again train on resized and scaled data, now from the relevant 12 665 elements of the training set and testing on the relevant 2115 elements of the test set, and utilize the small-angle map (5) with $a = 0.1$. Unsurprisingly, a modest cutoff of $\varepsilon = 2 \times 10^{-4}$ resulting in a maximum bond dimension of 79 is sufficient to get a near-perfect average F_1 score of $\bar{F}_1 = 0.9985$ on the test data set. We now turn to the evaluation of the model when only partial data are utilized, as described in Sec. IID. Figure 7 shows the results when only a subset of data rows or columns is utilized, with the ordering of the utilized data as shown. The results for a single row or column are not shown due to there being a “border” around the images that results in no useful information being present. We see a rapid increase in \bar{F}_1 as

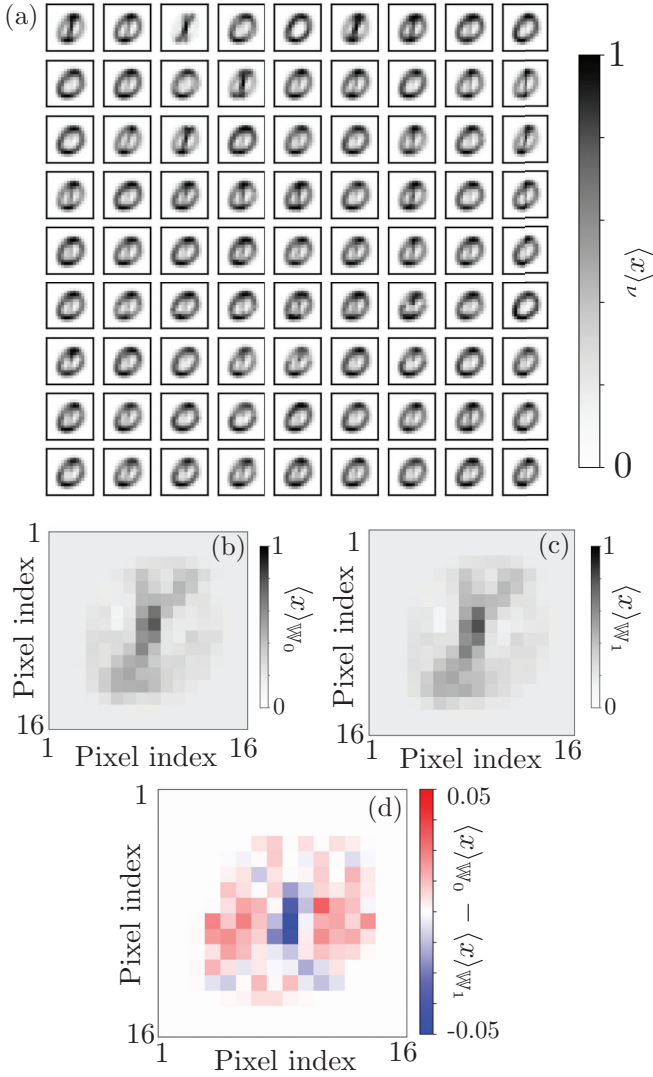


FIG. 8. One-point averages of classification weight vectors at the data scale. The data-scale representation of the one-point averages of vectors at the highest level of scale is displayed as images. (a) Representations of all 81 feature vectors extracted by the TTN, showing their mixed nature between 0 and 1 due to the unsupervised procedure. Data-scale representations of the weight vectors are shown for (b) class 0 and (c) class 1. While they look very similar, their difference, shown from a close-up view in (d), reveals that the weight vector for class 1 is higher in the central region and the weight vector for class 0 is higher towards the edges.

more data are included until the results essentially saturate at the score given by the full data array, occurring near the point where half of the data is used for classification. We note that more accurate results with partial data may be possible by using an alternate training strategy that utilizes a cost function involving partial data vectors.

We now investigate the interpretability of the decision weight vectors using the methodology developed in Sec. II E. For the purposes of having a square number of feature vectors for visualization, we use a cutoff of $\varepsilon = 2 \times 10^{-4}$ on all layers up to the last, in which a bond dimension of 81 is used. In Fig. 8(a) we show the 81 feature vectors that form the basis

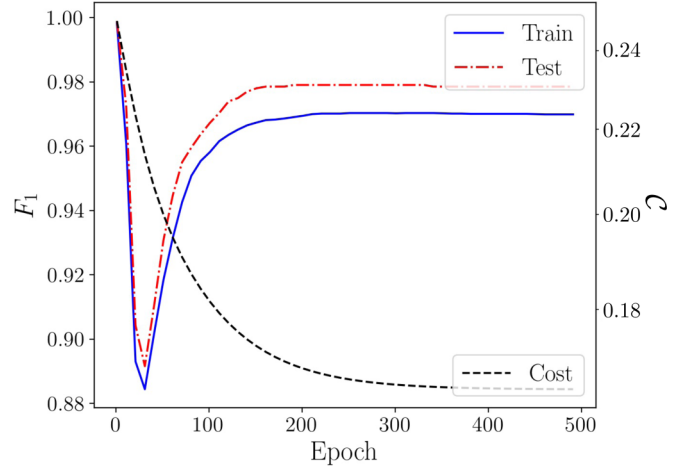


FIG. 9. Evolution of \bar{F}_1 score and cost function during isometric optimization: the training (blue solid line) and test (red dot-dashed line) average F_1 scores (left axis) and the cost function C from Eq. (14) (black dashed line, right axis) during the isometric optimization process of Eqs. (27)–(29) with $\beta = 0.1$ and $\eta = 0.1$. The starting point for optimization is the nearest isometric matrix to the unconstrained optimum, defined in Eq. (32).

for the decision space at the highest level of scale, visualized by fine graining to the data scale, taking one-point averages and displaying as an image. Here we see that the unsupervised feature extraction procedure results in feature vectors whose character is a mixture of the 0 and 1 digits, with some vectors having a more 0-like character and others having a more 1-like character. Using the particular linear combination of feature vectors specified by the unconstrained weight vectors (15) for the 0 class and 1 class, we find the one-point averages shown in Figs. 8(b) and 8(c), respectively. Visually, we see little difference between the two vectors, with both having features of 0-like and 1-like character. To see how the two weight vectors define a classification boundary, we examine the difference of their one-point averages on a close-up scale in Fig. 8(d). We observe that the weight vector for class 0 is slightly larger in a ring surrounding the central pixels, while the weight vector for class 1 is largest on the central pixels. The center pixel in particular has the largest difference for the two classes, indicating that this particular pixel is key for separating the two digits with the given weight vectors.

In the above we have utilized the ideal weight vectors that minimize the quadratic cost function (14) defining our supervised classification problem. We now look at optimizing an isometric set of weight vectors that define a mapping from the renormalized feature vectors into a class decision qubit. In Fig. 9 we show the convergence behavior of the cost function during the manifold gradient descent procedure [Eqs. (27)–(29)], starting from the nearest isometric matrix defined in Eq. (32) and utilizing the parameters given in the figure. As a point of reference, the unconstrained nonisometric weights had a cost function of approximately 4×10^{-6} . Also, while the unconstrained optimization resulted in training and test F_1 scores of 0.998 61 and 0.998 53, the nearest isometric weights have scores of 0.998 23 and 0.998 97 and the results following 500 epochs of isometric optimization with the parameters

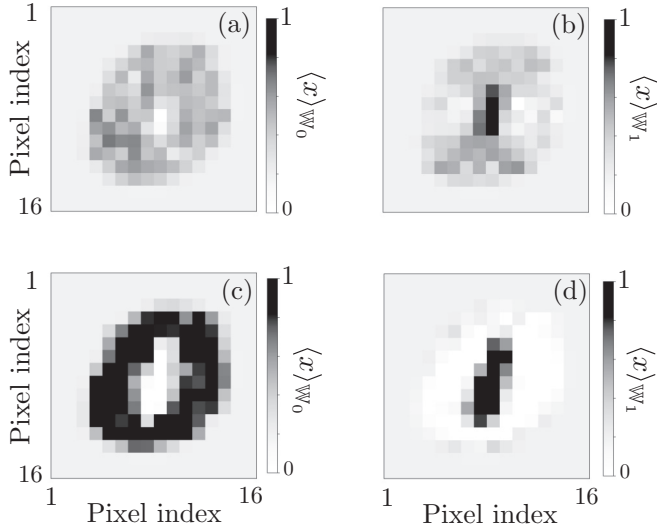


FIG. 10. One-point averages of isometric classification weight vectors at the data scale. (a) and (b) Analogs of Figs. 8(b) and 8(c), utilizing the nearest isometric weight vectors to the unconstrained optimum [see Eq. (32)]. (c) and (d) Corresponding analogs following 500 epochs of manifold gradient descent as in Fig. 9. As gradient descent progresses, the corresponding data-scale averages become further distinguishable.

in Fig. 9 are 0.969 92 and 0.978 56. As expected, the cost function shows a monotonic decrease during optimization, but the F_1 scores show a nonmonotonic behavior.

With the isometric weights in hand, we now revisit the interpretability analysis from above, starting with a weight vector and fine graining back to the data scale. We note that the choice of weight vectors does not affect the interpretability of the coarse-grained feature vectors shown in Fig. 8(a), but only their weighting for the final class decision. The data-scale representation of the one-point averages of the isometric weight vectors is shown in Fig. 10, with Figs. 10(a) and 10(b) corresponding to the nearest isometric weight vectors to the unconstrained optimum (32) and Figs. 10(c) and 10(d) corresponding to the weight vectors following 500 epochs of manifold gradient descent with the parameters of Fig. 9. Comparing with Fig. 8, we can clearly see that the orthogonality constraint has resulted in more human-interpretable features and has spread more of the decision importance throughout the data array rather than concentrating it into a few pixels, with both characteristics becoming more prominent following manifold optimization. This motivates the use of isometric weights as a form of model regularization that may make the results less susceptible to adversarial perturbations, a key consideration for both classical [74,75] and quantum [76,77] machine learning. The use of orthogonality constraints in classical deep learning architectures has also been explored (see, e.g., Ref. [78]).

As mentioned in Sec. II E, our interpretability metrics are not limited to one-point averages. In Fig. 11 we investigate two metrics of two-point correlations extracted from the weight vectors fine grained to the data scale. Here we fix the location of one of the pixels to be the central location c and display correlation metrics as a function of the other pixel

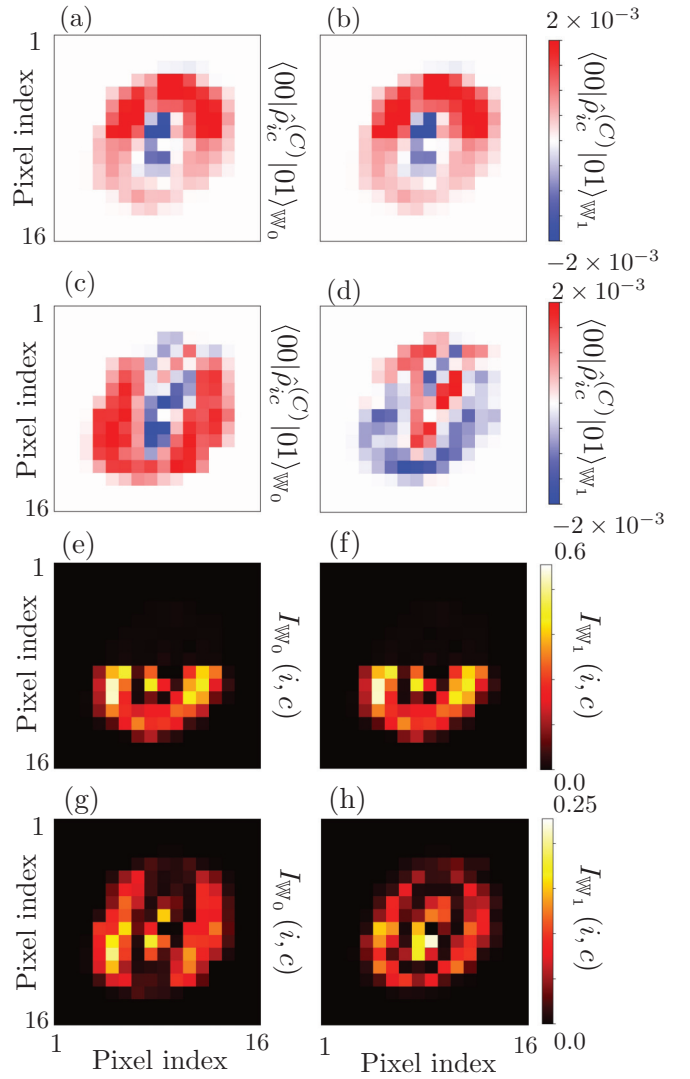


FIG. 11. Two-point correlations of classification weight vectors at the data scale. Elements $\langle 00 | \hat{\rho}_{ic}^{(C)} | 01 \rangle$ of the correlation density matrix obtained from the unconstrained classification weights are shown for (a) class 0 and (b) class 1. (c) and (d) Analogs for the isometric classification weights following gradient descent. Mutual information of the unconstrained weight vectors is shown for (e) class 0 and (f) class 1, evaluated between the center pixel and each of the other pixels in the array. (g) and (h) Analogs for the isometric weight vectors following gradient descent. The use of isometric data structures separates the two class decision vectors in the data space compared with the nonisometric case and better utilizes correlations at the highest level of scale, resulting in more uniform values in the top and bottom halves of the arrays.

location as an image. Figures 11(a)–(d) use $\langle 00 | \hat{\rho}_{ic}^{(C)} | 01 \rangle$, a coherence of the correlation density matrix between pixels c and i [Eq. (35)], which is a measure of correlation when positive and anticorrelation when negative. Figures 11(a) and 11(b) are for the unconstrained weight vectors following normalization. Similar to Fig. 8, very little distinction is seen between the data-scale metric for the class 0 and class 1 weight vectors. Another clear characteristic of these panels is that the correlation is different for the upper and lower

halves of the image. Given that the final tensor in the feature extractor encodes correlations between the upper and lower halves of the pixel array (see Fig. 5), this indicates that the weight vectors are not significantly utilizing correlations at the highest level of scale. We can contrast this with Figs. 11(c) and 11(d), which are the same metrics for the isometric weight vectors following gradient descent. These correlations show the same order of magnitude on the top and bottom of the pixel array, demonstrating that correlations are weighted similarly at all scales, and we see clear patterns of correlation and anticorrelation that are different between the class 0 and class 1 weight vectors. Figures 11(e)–(h) use the metric of mutual information [Eq. (37)], with Figs. 11(e) and 11(f) being the (normalized) unconstrained weight vectors and Figs. 11(g) and 11(h) being the isometric weight vectors following gradient descent. Similarly to the correlation density matrix metric, the unconstrained weight vectors show no visible difference between class 0 and class 1 and show a higher degree of correlation in the lower half of the pixel array (containing the fixed pixel location c) than in the upper half. The isometric weight vectors in Figs. 11(g) and 11(h) show similar orders of magnitude of mutual information in the upper and lower halves of the pixel array and distinct patterns for the two classes.

B. Human activity recognition

In order to demonstrate applicability of TTN-based classifiers to multivariate time-series data, we also apply the above methods to a data set for human activity recognition (HAR) [72]. The data set is formed from smartphone accelerometer and gyroscope data recorded for 15-s intervals at 50 Hz from 30 subjects. The activities comprise six classes: walking, walking upstairs, walking downstairs, sitting, standing, and laying. The data were further processed using a median filter and a third-order low-pass filter with 20-Hz cutoff frequency to reduce noise, and an additional low-pass filter with a cutoff frequency of 0.3 Hz was utilized to remove gravitational forces from the accelerometer data. Finally, the data were resampled in sliding windows of 2.56 s with 50% overlap to result in 128 data points per time series. The training (test) data set consists of 7352 (2947) collections of time series.

We will focus on the task of classifying a time series as being either walking (i.e., coming from the walking, walking upstairs, or walking downstairs classes of the original data set), which we will refer to as class 0, or not walking (coming from the sitting, standing, or laying classes of the original data set), which we will refer to as class 1. We then utilize the norms of the total acceleration vector $a(t) = \sqrt{a_x^2(t) + a_y^2(t) + a_z^2(t)}$ and total angular velocity vector $g(t) = \sqrt{\omega_x^2(t) + \omega_y^2(t) + \omega_z^2(t)}$ as input features to the map (5) feeding the TTN feature extractor. The final stage in processing is to rescale all data to the intervals $a(t) \in [0, 1]$ and $g(t) \in [0, 1] \forall t$, which also renders the data dimensionless. The data in the training set, processed as described above, are shown for the two classes in Fig. 12, with Figs. 12(a) and 12(b) displaying the time series for $a(t)$ and Figs. 12(c) and 12(d) displaying the time series for $g(t)$.

We apply a TTN feature extractor to the full time series, with the data ordering $\{a(1), g(1), a(2), g(2), \dots, a(L), g(L)\}$

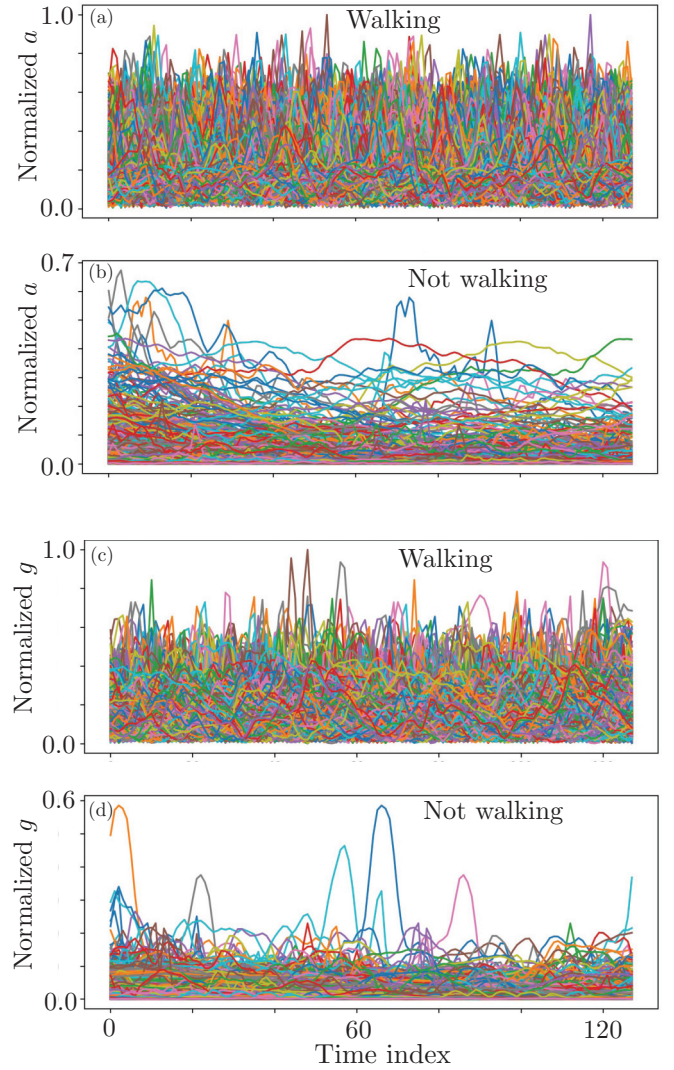


FIG. 12. Derived time-series features for the HAR data set. The dimensionless rescaled acceleration vector norms [(a) class 0 and (b) class 1] and angular velocity vector norms [(c) class 0 and (d) class 1] for the data in the training data set are displayed as a function of the time index. While individual trajectories are not resolved in this view, statistical differences between the training classes are visible.

such that the first layer mediates correlations a and g at the same time and higher layers facilitate correlations between all features at different times. Using this topology, a cutoff of $\varepsilon = 1 \times 10^{-6}$, and a maximum bond dimension of 812, we find training and test average \bar{F}_1 scores of 0.9974 and 0.9627, respectively, with the weight vectors found by unconstrained optimization (15). In Fig. 13 we show the results of our average one-point interpretability analysis (34) for the class weight vectors in the two classes and for the two features a and g , together with the differences between the time series for the two cases. Similar to what was seen in the example using MNIST image data shown in Fig. 8, very little difference can be seen between the one-point averages at the data scale, here interpreted as time series. In contrast to the MNIST example, in which a more detailed analysis of the difference of the

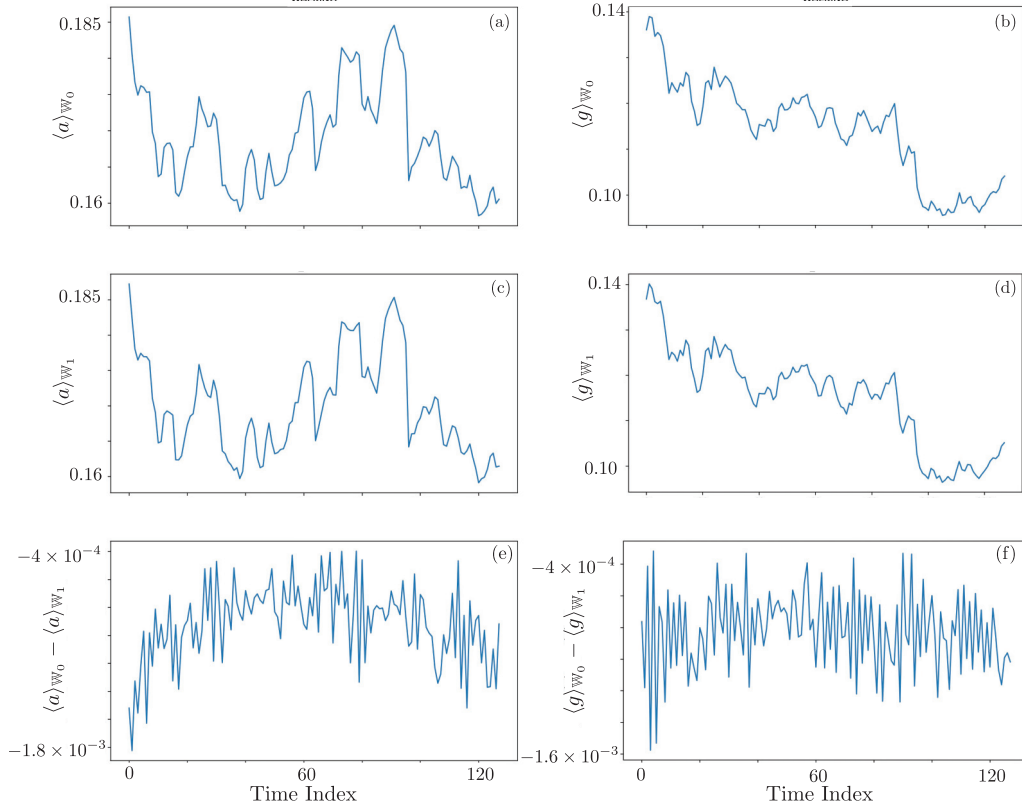


FIG. 13. Average one-point representation of classification weight vectors at the data scale. The data-scale representations of the class decision weight vectors are shown for (a) and (b) class 0 and (c) and (d) class 1, with their difference in (e) and (f), shown for (a), (c), and (e) the acceleration vector norm feature and (b), (d), and (f) the angular velocity vector magnitude feature. Analogous to the MNIST example in Fig. 8, very little difference is discernible between the two classes from the one-point averages.

one-point averages identified data regions that were critical to the class decision, in the present case the difference of the time series displays significant variation across the range of the data and there is not an immediate interpretation of the data used in forming the class decision.

We now investigate the case in which the class decision weight vectors are restricted to form an isometric matrix, obtained by manifold gradient descent. We find a final cost function of 0.1729, which should be compared with the cost function 6.06×10^{-3} found by unconstrained optimization, following approximately 1000 epochs of manifold gradient descent with a final gradient norm of approximately 4×10^{-5} . The final training and test average \bar{F}_1 scores are 0.9418 and 0.9983, respectively. The difference in performance is almost completely due to data elements from class 0 being misclassified as class 1; the training data set has 400 misclassified elements of class 0 while the test data set properly classifies all elements from class 0. With the isometric class weight vectors in hand, we now repeat the one-point interpretability analysis shown in Fig. 13. The results are shown in Fig. 14. An immediately striking feature of the interpretability analysis is that the one-point average time series for class 1 are negative, while the definition of the input features as norm vectors requires physical features to be positive. An interpretation of this can be traced back to Eq. (34), where the one-point average is extracted. A negative value of the time series implies that the two amplitudes of the qubit encoding at the lowest

level of scale have opposite signs. To understand how this affects the class decision, we can define a mean-field weight vector for class c at the lowest level of scale as

$$|\Phi_{\text{MF};c}\rangle = \otimes_{j=1}^L \left(\sum_{i_j=1}^2 \phi_{i_j}^{(j)}(\langle x_j \rangle_{\mathbb{W}_c}) |i_j\rangle \right). \quad (41)$$

The dot product with a data vector \mathbf{z} at this level of scale is

$$\langle \Phi^{(0)}(\mathbf{z}) | \Phi_{\text{MF};c} \rangle = \prod_{j=1}^L \phi^{(j)}(z_j) \phi^{(j)}(\langle x_j \rangle_{\mathbb{W}_c}) \quad (42)$$

$$= \prod_{j=1}^L \cos\left(\frac{a}{x_{\max}}[z_j - \langle x_j \rangle_{\mathbb{W}_c}]\right). \quad (43)$$

Recalling that $\frac{a}{x_{\max}} \ll 1$, each of the cosines can be expanded as

$$\cos\left(\frac{a}{x_{\max}}(z_j - \langle x_j \rangle_{\mathbb{W}_c})\right) \approx 1 - \frac{a^2}{2x_{\max}^2}(z_j - \langle x_j \rangle_{\mathbb{W}_c})^2. \quad (44)$$

Hence, the contribution from the component of test data vector \mathbf{z} at time point j , z_j , to the class decision is given roughly by its distance from the average time-series vectors for the two classes at this time point $\langle x_j \rangle_{\mathbb{W}_c}$. While the test data elements z_j will never take on negative values, they may be closer to small negative values than large positive values, and the appearance of negative values in $\langle x_j \rangle_{\mathbb{W}_c}$ may facilitate

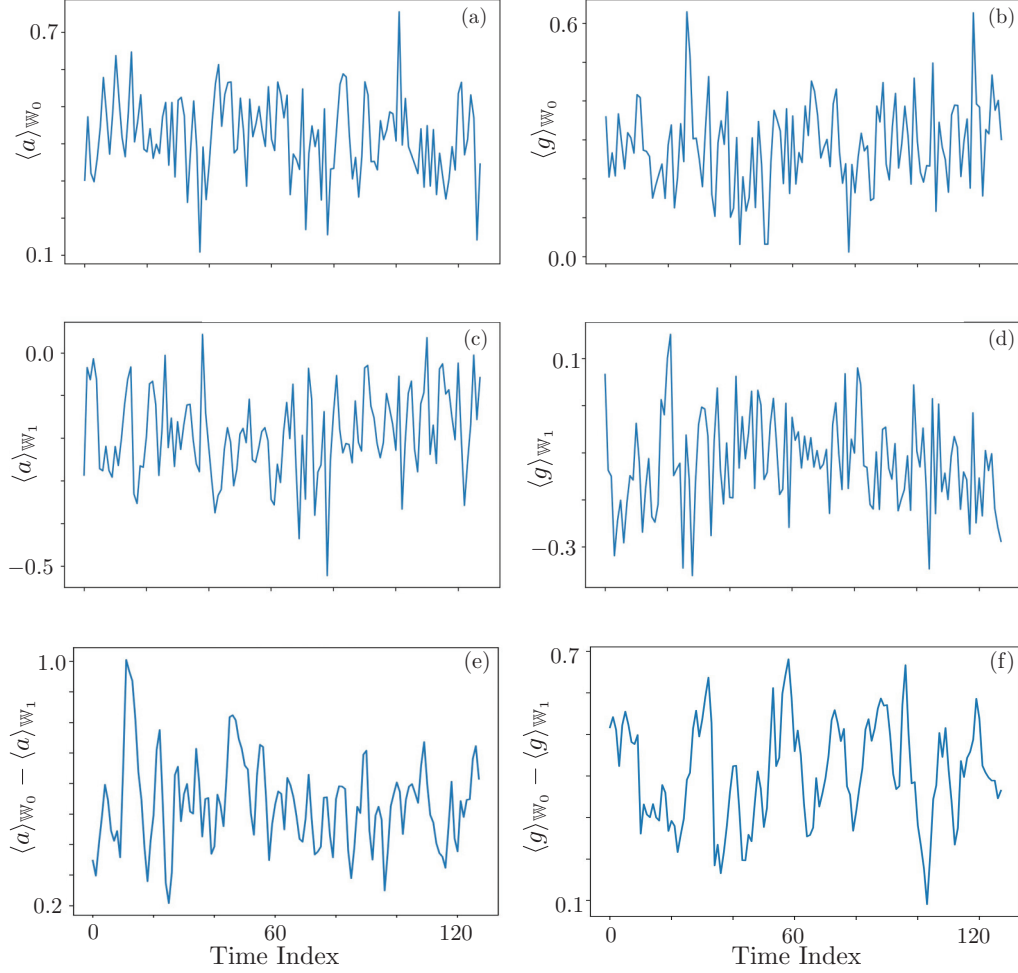


FIG. 14. Average one-point representation of isometric classification weight vectors at the data scale. The data-scale representations of the isometric class decision weight vectors are shown for (a) and (b) class 0 and (c) and (d) class 1, with their difference in (e) and (f), shown for (a), (c), and (e) the acceleration vector norm feature and (b), (d), and (f) the angular velocity vector magnitude feature. The isometric constraint separates the time series for the two classes significantly compared with the unconstrained case in Fig. 13.

orthonormality between the weight vectors in the various classes.

The above one-point analysis in Eqs. (41)–(44) does not account for correlations between the a and g features, or between features at different times, and so does not fully capture all of the information present in a correlated quantum model. To look at correlations beyond one-point averages, we look at the two-point mutual information between acceleration and angular velocity features at different times, displayed in Fig. 15. Here Figs. 15(a), 15(c), and 15(e) [Figs. 15(b), 15(d), and 15(f)] show the mutual information of the classification decision weight vector for class 0 (class 1) and Figs. 15(a) and 15(b), Figs. 15(c) and 15(d), and Figs. 15(e) and 15(f) show correlations between accelerations, cross correlations between accelerations and angular velocities, and correlations between angular velocities at different times, respectively. The weight vectors display a complex set of correlation behaviors, with correlations seen between all features spread across timescales at comparable magnitudes. However, we do find that the strongest correlations are those between the acceleration features at nearby times.

IV. COMPILATION FOR A QUANTUM COMPUTER

So far, our discussion has been focused on the training and deployment of a machine learning model utilizing quantum data structures on a classical computer. However, one of the key features of tensor networks is that they correspond to low-depth quantum circuits and so provide a natural blueprint for compilation onto a quantum device [24,27]. In this section we will provide some detail into a process for taking a classically trained TTN model and converting it into a sequence of quantum operations to be applied on a quantum computer. We will refer to this process as quantum compilation. We utilize techniques similar to those applied to compiling matrix product state generative models [49], in which the inherent freedom of the TN structure is utilized to aid in compiling to quantum hardware and greedy heuristics are used to compile isometries into native operations on target hardware. This stage of quantum compilation enables optimal classical optimization strategies to be utilized to precondition a quantum model, defining a model architecture and initial guesses at parameters for the gate set and topology of a given device. Also, as we

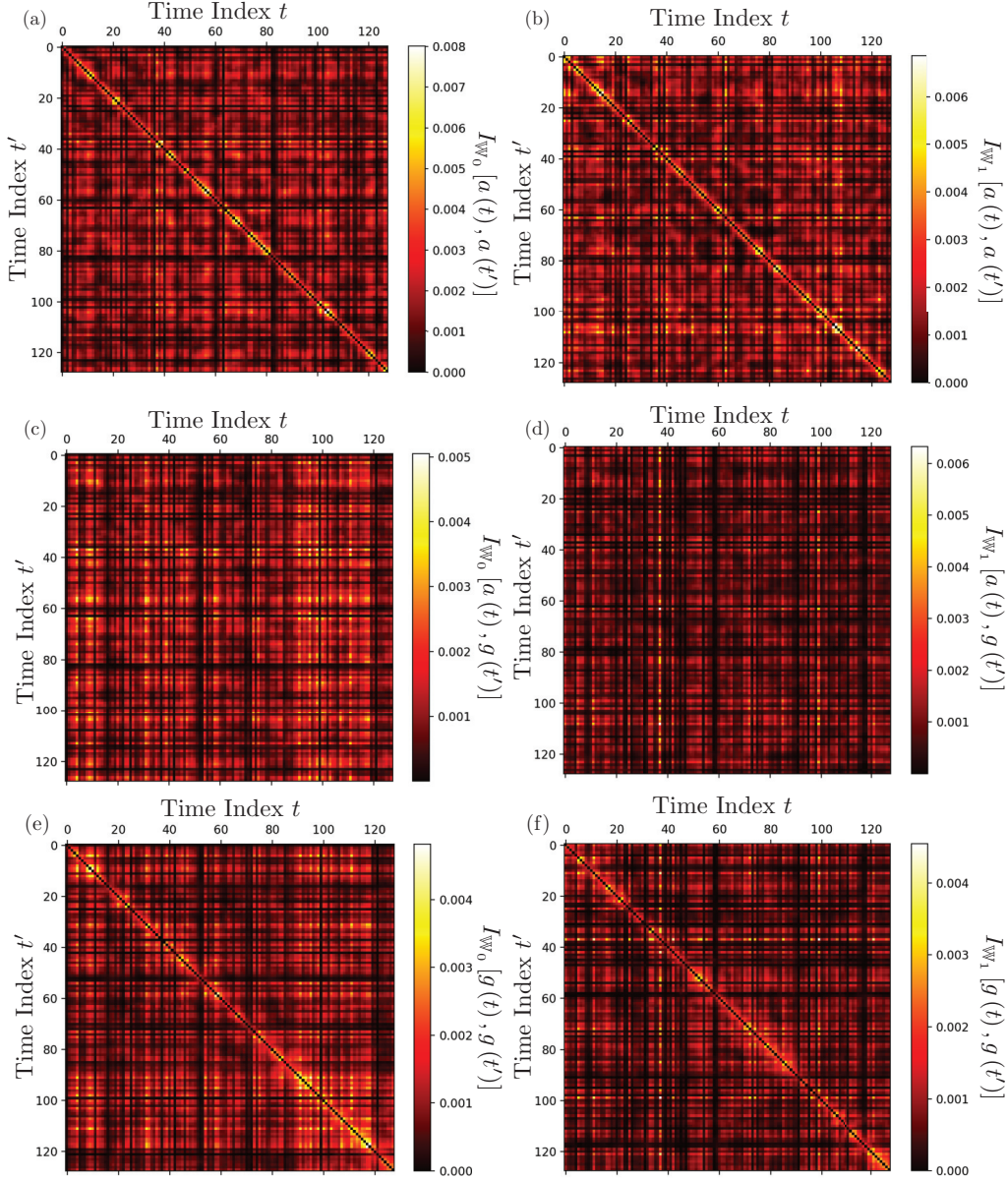


FIG. 15. Two-point mutual information of isometric classification weight vectors at the data scale. The mutual information metric (37) of the isometric classification decision weight vectors for (a), (c), and (e) class 0 and (b), (d), and (f) class 1 are shown as functions of time index, for (a) and (b) correlations between acceleration features at different times, (c) and (d) cross correlations between accelerations and angular velocities at different times, and (e) and (f) correlations between angular velocities at different times. In all cases, correlations are seen between features at all timescales, with the strongest correlations seen between acceleration features at nearby times.

will show, the compiled architecture can also be utilized as an *Ansatz* that is not subject to the isometric constraints of the classical model and optimized in its own right. In particular, this architecture can be further refined directly on the quantum device using a hybrid quantum-classical optimization strategy [79–83]. To demonstrate our methods concretely, throughout this section we will consider a TTN classifier that operates on an eight-dimensional classical feature vector, as shown in Fig. 16(a), and encodes a binary class decision in a two-dimensional Hilbert space represented by the free line at the top of the network in Fig. 16(a).

A. Quantum compilation of isometric tree tensor networks with postselection

The key operation in the TTN feature extractor is the application of an isometry $U_{\mu_\xi v_\xi}^{\mu_{\xi+1}}$ at the level of scale ξ to two feature vectors $|\phi_{\mu_\xi}\rangle$ and $|\psi_{v_\xi}\rangle$ to produce a new feature vector $|\Xi_{\mu_{\xi+1}}\rangle$ at the level of scale $(\xi + 1)$. Assuming that the states at level ξ can be described by a χ_ξ -dimensional basis of states, they can each be encoded into a quantum register of $\log_2 \chi_\xi$ qubits. Hence, each is an operation on $2 \log_2 \chi_\xi$ qubits that produces a result on $\log_2 \chi_{\xi+1}$ qubits with any remainder being decoupled so that they can be reinitialized and hence

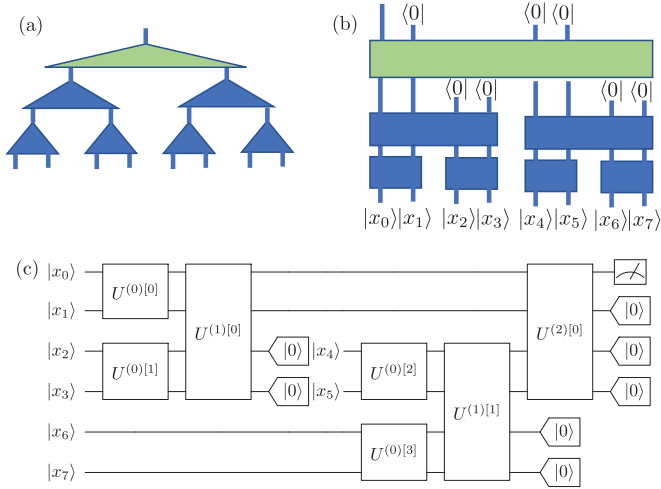


FIG. 16. From an isometric tensor network to a quantum circuit with postselection. (a) A TTN structure with a feature extractor acting on an eight-dimensional classical data vector to produce a binary class decision encoded in the two-dimensional Hilbert space represented by the top line. Each triangle represents an isometry which can be obtained by classical optimization. (b). Quantum-tensor-network representation of the TTN in (a) for $\chi = 4$. Each box is a unitary operation which has been chosen to match the isometries from (a) where they have nonzero support. Lines ending in $|0\rangle$ denote decoupled qubits which require postselection to ensure the isometries from (a) have been applied. (c) Quantum circuit pseudocode for the quantum TTN from (b), utilizing MCMR to define the model on six qubits. Tags containing $|0\rangle$ denote postselection and the topmost qubit measurement defines the class decision.

reutilized in the computation. For lower layers of the TTN in which $\chi_\xi^2 \leq \chi$, the corresponding isometries $U^{(\xi)}$ are in fact unitaries. This is demonstrated in Fig. 16(b), in which each of the lines denotes a two-dimensional Hilbert space (i.e., a qubit) and the boxes denote unitary operators taking the group of indices on the bottom to the group of indices on the top. Unitarity is seen in this representation by the fact that all lines of the tensors on the lowest layer connect to the next layer of tensors.

For higher layers in which $\chi_\xi^2 > \chi$, the operator $U_{\mu_\xi \nu_\xi}^{\mu_{\xi+1}}$ is an isometry and not all states are passed up to the next layer. This has two important consequences for quantum compilation. The first is that this operation no longer fully specifies a unitary operation, as is required for implementation on a quantum computer. While this can be remedied in a variety of ways, here we adopt the methods of Ref. [49] in which a greedy algorithm produces unitaries of increasing gate depth which are optimized according to their distance with the nonzero elements of the target isometry. This method will produce unitaries with short gate depths which match the desired isometry where the latter has support and also can readily account for restrictions on near-term devices such as limited native gate sets, connectivity, and hardware noise. This same procedure remedies the issue in which $\chi_{\xi+1}$ is not a power of 2 and so does not fully specify an operation on a set of hardware qubits. We note that there is an entire family of unitaries which will match the target isometry to the desired tolerance and our procedure uses this freedom to find unitaries of as short of

gate depth as possible; this is a key consideration for noisy near-term devices.

The second consequence of dealing with isometric tensors is that the particular representation of the isometry denotes a state in which some of the qubits decouple and are not passed to the next layer. While there is freedom in choosing the decoupled state of these qubits (e.g., by an additional unitary transformation on the decoupling subspace), we will follow the standard convention that all decoupled qubits are placed in the $|0\rangle$ state. The decoupled qubits are represented in Fig. 16(b) by lines terminating in a state $|0\rangle$. In order to clarify the impact of qubit decoupling on a TTN classifier, it is useful to first describe its impact on a generative circuit given by the adjoint tensors. Namely, the generative interpretation of an isometry $\hat{U}^{(\xi)}$, in which the adjoint isometry $\hat{U}^{(\xi)\dagger}$ is applied to the state $|\mu_{\xi+1}\rangle|0 \dots 0\rangle$ to produce $|\mu_\xi\rangle|\nu_\xi\rangle$ at a lower level of scale, underlies the sequential preparation of tensor-network states utilizing ancilla qubits [24–26,49]. Here, as before, $|0 \dots 0\rangle$ denotes the decoupled qubit state. Following this generative interpretation of the full TTN circuit starting from the state $|\ell\rangle|0 \dots 0\rangle$ encoding class label ℓ and applying the adjoints of all tensors in the TTN will produce a representative feature vector at the lowest level of scale. Hence, this is the quantum version of the fine-graining procedure described in Sec. II E.

In the discriminative interpretation where we start from the lowest level of scale and work upward, the appearance of decoupled qubits hence implies postselection, where the action of the desired isometry is indicated by measuring the decoupled qubits to be in the state $|0\rangle$ following circuit execution. If another state of the decoupled qubits is measured, the action of the circuit does not correspond to the action of the isometry described by the original (classically optimized) TTN model, but instead corresponds to the complement of the isometry which defines the remainder of the unitary required for implementation on a quantum computer. The correspondence between the generative interpretation and the fine-graining procedure of the preceding paragraph sheds light on the expected success probability of postselection for the discriminative model: Roughly, this should scale as the volume of the states encoded by the TTN feature extractor (and sampled in the generative interpretation) divided by the volume of the feature vectors defining the training and test sets. For well-constructed data sets and TN models of high fidelity, we can expect this success probability to be high. Nevertheless, the fact that isometric TN models generally require postselection will motivate us to consider models that do not require postselection momentarily. As we will see, the isometric models are still useful in this case, as they enable data-driven classical preconditioning of the general model architecture.

A quantum circuit pseudocode description of the isometric $\chi = 4$ model with a classical data vector of length 8 is shown in Fig. 16(c). Here the tags containing the state $|0\rangle$ indicate postselection. This circuit uses midcircuit measurement and reuse (MCMR), as has been demonstrated in trapped ion hardware [84], to reduce the qubit number requirement for implementation. In general, a TTN feature extractor with bond dimension χ acting on an L -dimensional classical data vector has qubit requirements of $\mathcal{O}(\log_2 L \log_2 \chi)$ with MCMR,

demonstrating that TTN models can be extremely resource efficient.

Before describing models that do not require postselection, we look at the final classification stage of the TTN network. In the case that the weight vectors $\{|W_\ell\rangle\}$ discussed in Sec. II C define an isometric matrix, the operation that takes the feature vectors output from the TTN extractor into a collection of $\log_2 C$ class decision qubits is of the same form as those for the feature extractor and so the same methods can be applied. Following application of this operation, the class decision qubits can be measured in the computational basis and the outcome is the predicted class index in a binary representation. In the case that the weight vectors do not form an isometric matrix, we can still extract the decision functions $f_\ell(\mathbf{x}) = \langle W_\ell | \Phi(\mathbf{x}) \rangle$ using a SWAP test as follows. We first couple an ancilla qubit to the χ -dimensional state at the highest level of scale to form the state $|\psi_\ell\rangle = [|0\rangle|W_\ell\rangle + |1\rangle|\Phi(\mathbf{x})\rangle]/\sqrt{2}$ and then measure the probability for the ancilla to be in the state $|\leftarrow\rangle = [|0\rangle - |1\rangle]/\sqrt{2}$. This probability is $|\langle\leftarrow|\psi_\ell\rangle|^2 = [1 - f_\ell(\mathbf{x})]/2$, from which the decision function can be extracted. In the often-encountered case that $|W_\ell\rangle$ is not of unit norm, the normalized state $|W_\ell\rangle/\sqrt{\langle W_\ell|W_\ell\rangle}$ can be used to form the state $|\psi_\ell\rangle$ and then the norm added back in postprocessing to extract $f_\ell(\mathbf{x})$. By performing C such measurements, one for each value of ℓ , all decision functions can be extracted and the results classically postprocessed for the class decision. Clearly, dealing with a set of weight vectors that does not form an orthonormal set requires more complex operations and additional overhead in quantum resources and number of circuit runs.

B. Quantum compilation of isometric tree tensor networks without postselection

In principle, defining a TTN model which does not require postselection is straightforward: We replace each instance of postselection with a partial trace over the corresponding qubit output from a unitary. This results in the tensor network shown in Fig. 17(a), which represents the reduced density matrix of the class decision qubit following action of the TTN circuit. We can see that the action of the TTN circuit amounts to a completely positive and trace preserving (CPTP) map applied to the pure density matrix obtained by the outer product of an encoded classical data vector with itself. Such models have been considered in Ref. [27] and classically optimized using the simultaneous perturbation stochastic approximation algorithm applied to the upper triangle of the Hermitian matrix H parametrizing a unitary as $U = \exp(iH)$.

When quantum compiling a model defined in terms of unitaries, we run up against the observation that most unitaries require deep circuits to compile [85]. Hence, it is advisable to operate with a shallow circuit description of the unitaries directly rather than optimizing the unitary and then attempting to compile. Significant research has been devoted to families of circuits with variable depth which can be variationally optimized. Here it has been shown that global optimization of randomly initialized circuits is subject to so-called barren plateaus in which gradients of the variational circuits vanish exponentially with increasing circuit depth and width, pre-

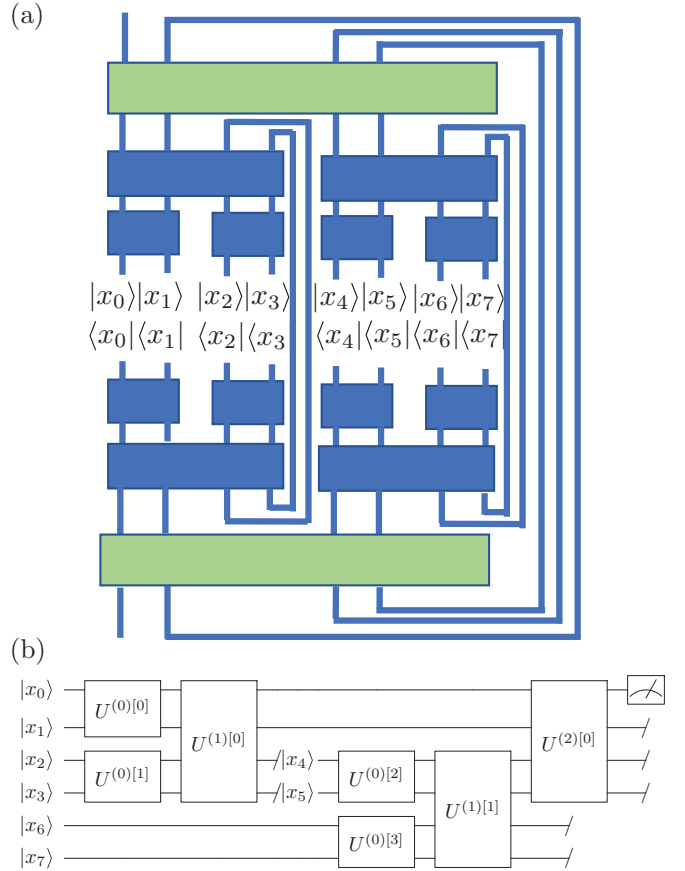


FIG. 17. From a tensor network to a quantum circuit without postselection. (a) A tensor network in which the uncontracted ancilla qubits are traced over acts as a CPTP map on the density matrix given by the outer product of an encoded feature vector with itself. The contracted network represents the reduced density matrix of the qubit encoding the class decision. (b) Quantum circuit without postselection whose measurement outcomes are given by the diagonal elements of the reduced density matrix in (a); compare Fig. 16 for the circuit with postselection.

cluding efficient training [86]. This is partially assuaged in our instance by the fact that the result is “local” in the sense that it is encoded in a single qubit or a small register of $\log_2 C$ qubits in the general multiclass setting [87]. Further gains in trainability can be expected by initializing the unitary away from a random guess [88], which can be thought of as a preconditioner. In the present situation, we advocate using the circuit and parameters obtained by compiling the postselected model as an initializer for the nonpostselected model, again replacing each instance of postselection with a trace. The parameters of this model can then be directly optimized, possibly directly on quantum hardware. As an example, Fig. 17(b) is the nonpostselected analog of the postselected circuit in Fig. 16(c), where a slash denotes reset of a qubit without postselection on its measurement. The benefit of using the postselected model as a preconditioner is that it only requires the compilation of isometries, and the freedom in how to represent the isometry as a unitary can be used to discover potentially shorter gate sequences in a data-driven fashion.

C. Example demonstration with HAR data

In this section we demonstrate the above procedures for quantum compilation with and without postselection on an example problem, again considering binary classification of the walking and not walking classes from the HAR data set described in Sec. III B. In order to reduce the size of the classical data vector, we do not work directly with the raw time-series data as in Sec. III B, but rather with features derived from it. The HAR data set itself [72] defines a set of 561 features derived from the time series; we downsample from these features by training a random forest classifier from SCIKIT-LEARN [89] and extracting the eight most relevant features from the data according to their ranked feature importance. These eight features define our classical data vector. We again use the map (5) with $a = 0.1$ to encode the data into quantum feature vectors, and train an isometric TTN and weight matrix with $\chi = 4$. Following optimization, we contract the weight vectors into the top layer of the feature extractor to form a $(\chi, \chi) \rightarrow 2$ tensor performing the combined operations of renormalization and mapping to a class decision. The topology of the resulting network corresponds to the tensor network in Fig. 16(a).

Similar to matrix product states (MPSs), TTNs have a gauge freedom in which a unitary matrix and its adjoint can be placed on any contracted line in the network without changing the overall state. For MPSs, this can be used to heuristically condition the isometries to reduce the expected gate depth for compilation [49]. Similar utilization of gauge freedom in TTNs is an interesting avenue for future research; here we use the gauge freedom only to fix signs such that the largest element in each row of the isometry is positive, where possible. We compile the isometries of the trained and conditioned TTN using the greedy procedure of Ref. [49] with a squared two-norm tolerance between the isometry and compiled unitary of $\varepsilon = 4 \times 10^{-4}$ and a set of controlled-NOT (CNOT) gates and R_y rotations. The lowest layer of tensors is all unitaries on two qubits, and we find that they can all be compiled to the desired tolerance with two CNOT gates. The next layer consists of four-qubit isometries, and we find that both tensors in this layer are compiled to the desired accuracy using 13 CNOT gates. The final layer is again an isometry on four qubits, and we find it is compiled to the desired tolerance using six CNOT gates. Explicit circuit representations are given in the Appendix. The test \bar{F}_1 score for the compiled postselected model is 0.9982.

As discussed in the preceding section, we now take the architecture of the compiled model with postselection and optimize its parameters based on a cost function for evaluation without postselection. For our chosen gate set of CNOT gates and R_y rotations, the parameters to be optimized are the rotation angles of the R_y operations. Following Ref. [27], we use the cost function

$$\mathcal{C} = \frac{1}{M} \sum_m \max(\rho_{\bar{\ell}_m \bar{\ell}_m} - \rho_{\ell_m \ell_m} + \lambda, 0)^\eta, \quad (45)$$

in which ℓ_m is the label of training instance m , $\bar{\ell}_m$ is the label of the incorrect class with highest model probability for training instance m , $\hat{\rho}$ is the reduced density matrix of the class decision qubit whose tensor-network representation is given in Fig. 17(a) and so $\rho_{\ell\ell}$ is the probability to

measure label ℓ from the circuit in Fig. 17(b), and λ and η are hyperparameters. We find that $\lambda = 0.2$ and $\eta = 2.0$ work well in this instance. We optimize the network using the limited memory Broyden-Fletcher-Goldfarb-Shanno algorithm implemented in PYTORCH [90] with a learning rate of 0.1. Following 30 epochs of optimization, we find a test \bar{F}_1 score of 0.9996. The explicit circuit representation is given in the Appendix. While it may seem surprising that the nonpostselected model has better performance than the postselected model, this is likely due to the fact that the postselected model utilized an unsupervised feature extraction layer while the nonpostselected model performed supervised optimization of the full network, as well as the relative simplicity of the chosen problem instance. In spite of this, our example demonstrates the potential utility for using isometric tensor-network models to obtain short-depth unitary circuits which can be reinterpreted as general CPTP maps and directly optimized, leading to efficient preconditioning of variational *Ansätze* in a principled and data-driven fashion.

V. CONCLUSION AND OUTLOOK

We have investigated classifiers based on an encoding of classical data vectors into quantum states in a Hilbert space that is exponentially large in the length of the data vector. An unsupervised feature extractor with a tree-tensor-network topology extracts a relatively small basis of relevant quantum states from a training set of data embedded into quantum states, with the size of this basis being the main hyperparameter of the model. The tensors in this network are optimized utilizing a procedure which keeps the dominant correlations between the quantum degrees of freedom encoding elements of the data vector, analogous to a renormalization group flow, at increasingly coarse levels of scale. The extracted feature vectors at the highest level of scale were utilized in a supervised cost function optimization to define a classification decision. We presented metrics for interpretability of such quantum classifiers which extract low-order correlations from the weight vectors interpreted as a quantum state and fine grained to the data scale by running the feature extraction network in reverse.

Building upon previous work on quantum-inspired algorithms using hierarchical tensor networks for classification, we utilized tools for the optimization of such models in which all elements correspond to proper quantum data structures that can be implemented on gate-based quantum computing devices. This included devising an embedding map with learning properties comparable to well-performing classical embedding maps, but which also produces valid quantum states. In addition, we utilized manifold-based optimization schemes that define an isometric mapping from the quantum feature vectors produced by the unsupervised feature extractor into a register of qubits whose probability amplitudes define a class decision. We discussed methods for translating the isometric tensors obtained through classical optimization into operations to be performed on a gate-based quantum computer and also discussed scaling of quantum resource requirements. Finally, we outlined a procedure for translating an isometric tensor-network model whose quantum implementation requires postselection into a preconditioned variational

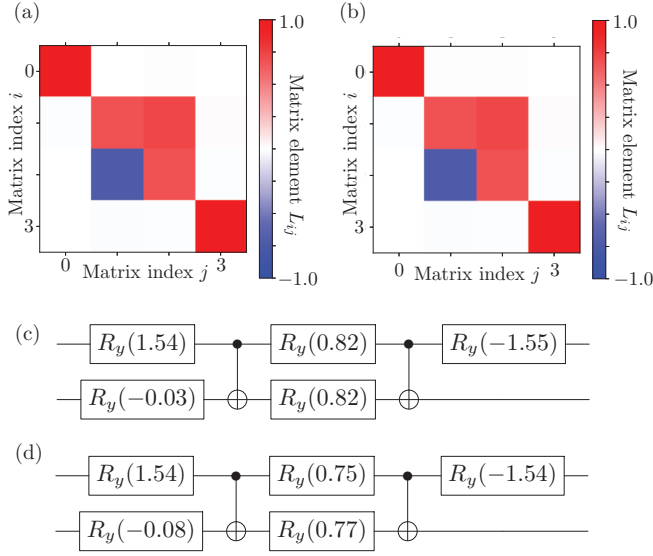


FIG. 18. Optimization for layer 0, tensor 0. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonposts-elected model.

quantum circuit *Ansatz* that does not require postselection and demonstrated that direct optimization of this latter structure can lead to models of comparable performance with the same circuit depth.

We applied the methods developed in this work on two data sets: the canonical MNIST database of handwritten digits and a multivariate time-series data set of human activity recognition. We demonstrated that a small-angle phase embedding map, which produces valid quantum states from classical data, gives comparable performance on the full MNIST data set to

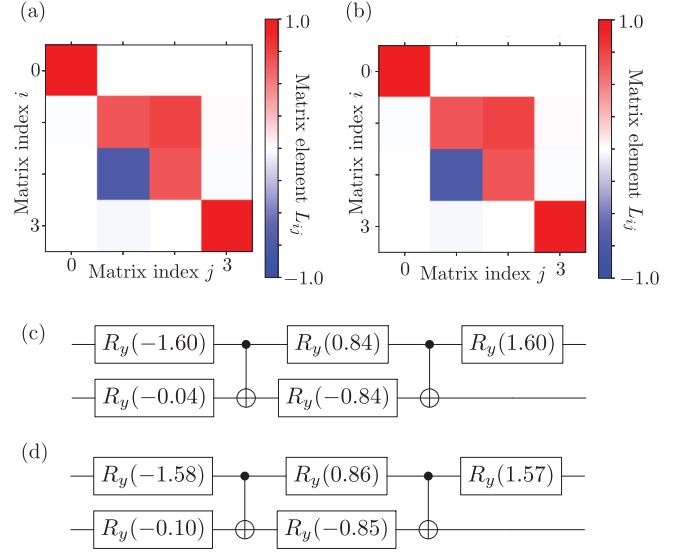


FIG. 20. Optimization for layer 0, tensor 2. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonposts-elected model.

the encoding of Ref. [56] mapping the data to a high-order polynomial. We then compared the results of a simpler classification problem of distinguishing the digits 0 and 1 from the MNIST data set using a unconstrained quantum-inspired model with those from the quantum model encoding the classification decision into the amplitudes of a qubit. We find that the use of fully quantum data structures produces more human interpretable features and utilizes more of the information and correlations across the data vector in making a classification decision, potentially improving robustness against noise or adversarial perturbations. Similar qualitative behavior was

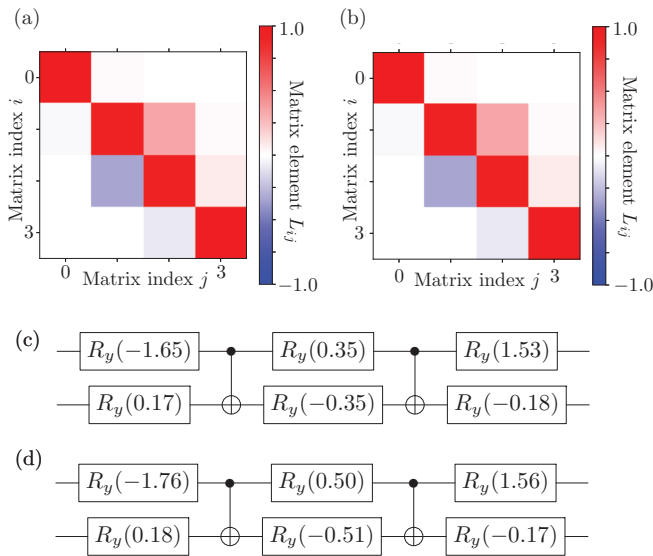


FIG. 19. Optimization for layer 0, tensor 1. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonposts-elected model.

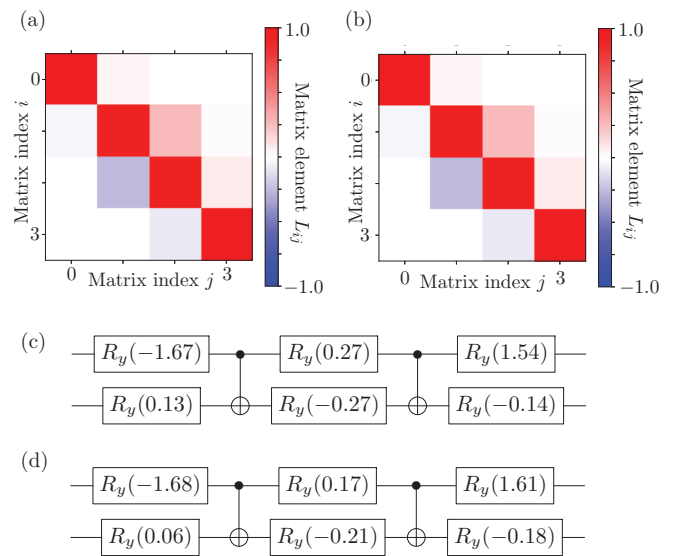


FIG. 21. Optimization for layer 0, tensor 3. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonposts-elected model.

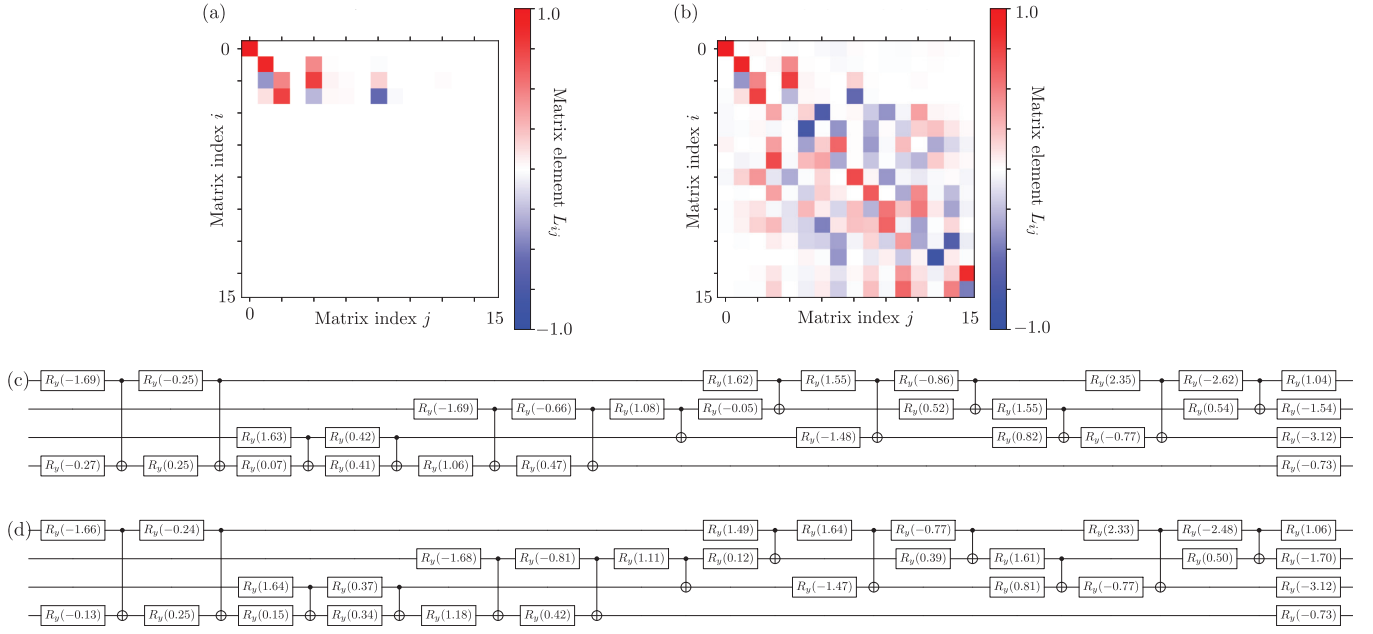


FIG. 22. Optimization for layer 1, tensor 0. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonpostselected model.

seen in the case of time series from the HAR data set, where we demonstrated the applicability of TTN-based classifiers to large-dimensional multivariate time series. A subset of features from the HAR data set was used to demonstrate our procedures for compiling classically optimized isometric TTN models for deployment on gate-based quantum computers, and for optimization of quantum models that do not require postselection.

Several opportunities exist for further research in quantum-assisted machine learning using tensor networks with a TTN

structure. For one, the tensors in our TTN feature extractor were defined using an unsupervised approach, but better performance may be had by using the isometric optimization methods described in this paper to the tensors of the feature extractor. More complex structures beyond trees, such as the multiscale entanglement renormalization *Ansatz* network [91] or combinations of trees and linear networks such as matrix product states [29,44], can be considered and can also be optimized using manifold gradient descent. Investigations of the fidelity of TTN models compiled to currently available

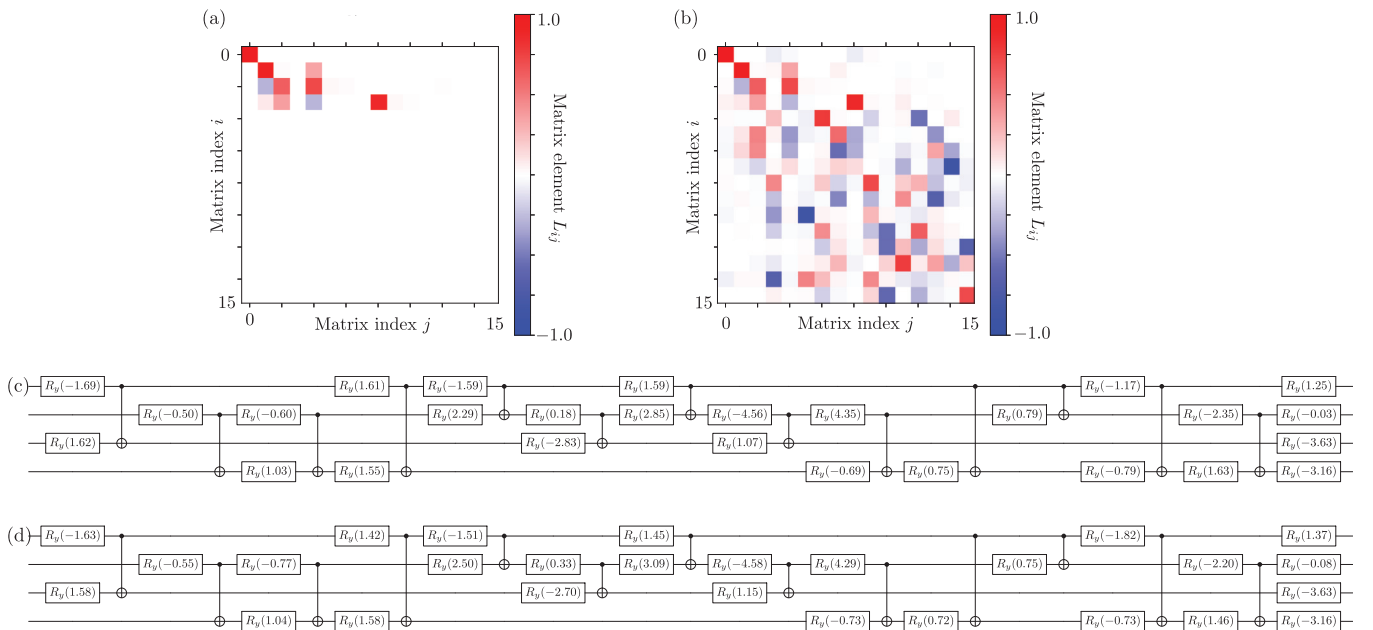


FIG. 23. Optimization for layer 1, tensor 1. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonpostselected model.

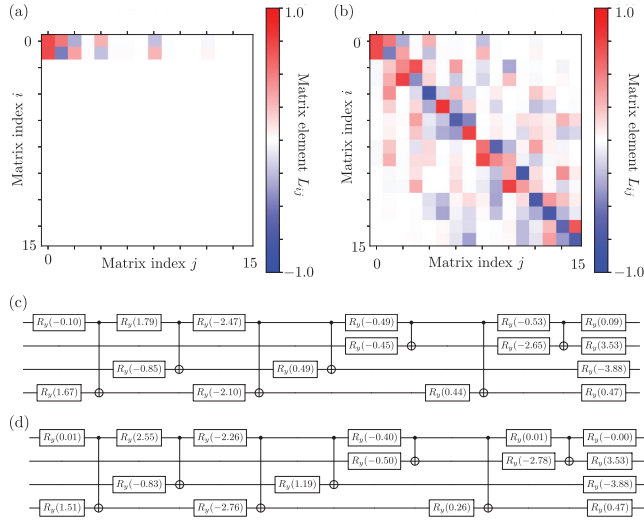


FIG. 24. Optimization for layer 2, tensor 0. (a) Isometry. (b) Compiled unitary for the isometry in (a). (c) Gate sequence for the unitary in (b). (d) Gate sequence for the unitary of the nonpostselected model.

quantum hardware and their resilience to noise, analogous to Ref. [49] for the case of generative matrix product state models, can be performed. Alternative training strategies, e.g., those that utilize partial data vectors or regularization terms, may provide better performance than the simple cost function utilized here. Finally, it is intriguing to consider methods for

the optimization of TTN-based classifiers with a quantum device in the loop, where performance can be investigated as the network is scaled to the classically intractable regime.

ACKNOWLEDGMENTS

We would like to thank M. Abernathy and G. Quiroz for useful discussions and acknowledge funding from the Internal Research and Development program of the Johns Hopkins University Applied Physics Laboratory.

APPENDIX: COMPILED GATES FOR THE HAR DEMONSTRATION

This Appendix contains the explicit gate sequences for the HAR demonstration of Sec. IV C in Figs. 18–24. Matrix plots display isometries or unitaries in the basis where significance increases with qubit index, i.e.,

$$\text{index}(|q_{N-1} \cdots q_0\rangle) = \sum_{i=0}^{N-1} 2^i q_i. \quad (\text{A1})$$

The panels of each figure display (a) the isometry defined by classical optimization, with whitespace for elements that are either 0 or undefined, (b) the compiled unitary representation of the isometry in (a), (c) the gate sequence for the unitary representing the compiled isometry in (b), and (d) the gate sequence for the unitary representing the nonpostselected model after optimization. Note that only the arguments in the rotations change in moving from panels (c) to (d).

- [1] P. W. Shor, *SIAM Rev.* **41**, 303 (1999).
- [2] C. Gidney and M. Ekerå, *Quantum* **5**, 433 (2021).
- [3] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, *Science* **309**, 1704 (2005).
- [4] B. Bauer, S. Bravyi, M. Motta, and G. K.-L. Chan, *Chem. Rev.* **120**, 12685 (2020).
- [5] E. T. Campbell, B. M. Terhal, and C. Vuillot, *Nature (London)* **549**, 172 (2017).
- [6] J. Preskill, *Quantum* **2**, 79 (2018).
- [7] R. S. Smith, M. J. Curtis, and W. J. Zeng, *arXiv:1608.03355*.
- [8] D. S. Steiger, T. Häner, and M. Troyer, *Quantum* **2**, 49 (2018).
- [9] T. Häner, D. S. Steiger, K. Svore, and M. Troyer, *Quantum Sci. Technol.* **3**, 020501 (2018).
- [10] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. Ben-Haim, D. Bucher, F. J. Cabrera-Hernández, J. Carballo-Franquis, A. Chen, C.-F. Chen, J. M. Chow, A. D. Córcoles-Gonzales, A. J. Cross, A. Cross, J. Cruz-Benito, C. Culver, S. D. L. P. González, E. D. L. Torre, D. Ding, E. Dumitrescu *et al.*, *Qiskit*, <https://qiskit.org> (IBM Research, Yorktown Heights, 2020).
- [11] R. LaRose, *Quantum* **3**, 130 (2019).
- [12] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell *et al.*, *Nature (London)* **574**, 505 (2019).
- [13] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, S. Boixo, M. Broughton, B. B. Buckley, D. A. Buell *et al.*, *Science* **369**, 1084 (2020).
- [14] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, *Nature (London)* **549**, 195 (2017).
- [15] A. Perdomo-Ortiz, M. Benedetti, J. Realpe-Gómez, and R. Biswas, *Quantum Sci. Technol.* **3**, 030502 (2018).
- [16] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, *Proc. R. Soc. A* **474**, 20170551 (2018).
- [17] I. Glasser, R. Sweke, N. Pancotti, J. Eisert, and J. I. Cirac, *Adv. Neural Inf. Proc. Syst.* **32**, 1498 (2019).
- [18] R. Sweke, J.-P. Seifert, D. Hangleiter, and J. Eisert, *Quantum* **5**, 417 (2021).
- [19] B. Coyle, D. Mills, V. Danos, and E. Kashefi, *npj Quantum Inf.* **6**, 60 (2020).
- [20] J. Tangpanitanon, S. Thanasilp, N. Dangniam, M.-A. Lemonde, and D. G. Angelakis, *Phys. Rev. Research* **2**, 043364 (2020).
- [21] U. Schollwöck, *Ann. Phys. (NY)* **326**, 96 (2011).
- [22] R. Orús, *Ann. Phys. (NY)* **349**, 117 (2014).
- [23] R. Orús, *Nat. Rev. Phys.* **1**, 538 (2019).
- [24] C. Schön, E. Solano, F. Verstraete, J. I. Cirac, and M. M. Wolf, *Phys. Rev. Lett.* **95**, 110503 (2005).
- [25] C. Schön, K. Hammerer, M. M. Wolf, J. I. Cirac, and E. Solano, *Phys. Rev. A* **75**, 032311 (2007).
- [26] D. Perez-Garcia, F. Verstraete, M. Wolf, and J. Cirac, *Quantum Inf. Comput.* **7**, 401 (2007).
- [27] W. Huggins, P. Patil, B. Mitchell, K. B. Whaley, and E. M. Stoudenmire, *Quantum Sci. Technol.* **4**, 024001 (2019).

- [28] E. Stoudenmire and D. J. Schwab, in *Proceedings of the 30th International Conference on Neural Information Processing Systems, Barcelona*, 2016, edited by D. D. Lee, U. von Luxburg, R. Garnett, M. Sugiyama, and I. Guyon (Curran, Red Hook, 2016), pp. 4799–4807.
- [29] E. M. Stoudenmire, *Quantum Sci. Technol.* **3**, 034003 (2018).
- [30] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, *npj Quantum Inf.* **4**, 65 (2018).
- [31] C. Guo, Z. Jie, W. Lu, and D. Poletti, *Phys. Rev. E* **98**, 042114 (2018).
- [32] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, *Nat. Mach. Intell.* **1**, 155 (2019).
- [33] G. Evenbly, [arXiv:1905.06352](https://arxiv.org/abs/1905.06352).
- [34] S. Klus and P. Gelß, *Algorithms* **12**, 240 (2019).
- [35] S. Cheng, L. Wang, T. Xiang, and P. Zhang, *Phys. Rev. B* **99**, 155131 (2019).
- [36] D. Liu, S.-J. Ran, P. Wittek, C. Peng, R. B. García, G. Su, and M. Lewenstein, *New J. Phys.* **21**, 073059 (2019).
- [37] I. Glasser, N. Pancotti, and J. I. Cirac, *IEEE Access* **8**, 68169 (2020).
- [38] T. Felser, M. Trenti, L. Sestini, A. Gianelle, D. Zuliani, D. Lucchesi, and S. Montangero, *npj Quantum Inf.* **7**, 111 (2021).
- [39] T.-D. Bradley, E. M. Stoudenmire, and J. Terilla, *Mach. Learn.: Sci. Technol.* **1**, 035008 (2020).
- [40] E. Gillman, D. C. Rose, and J. P. Garrahan, [arXiv:2002.05185](https://arxiv.org/abs/2002.05185).
- [41] J. Miller, G. Rabusseau, and J. Terilla, Tensor networks for probabilistic sequence modeling, *International Conference on Artificial Intelligence and Statistics* (PMLR, 2021), pp. 3079–3087.
- [42] R. Selvan and E. B. Dam, Tensor networks for medical image classification, *Medical Imaging with Deep Learning* (PMLR, 2020), pp. 751–732.
- [43] J. Wang, C. Roberts, G. Vidal, and S. Leichenauer, [arXiv:2006.02516](https://arxiv.org/abs/2006.02516).
- [44] J. A. Reyes and E. M. Stoudenmire, *Mach. Learn.: Sci. Technol.* **2**, 035036 (2021).
- [45] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, *Phys. Rev. X* **8**, 031012 (2018).
- [46] S. Efthymiou, J. Hidary, and S. Leichenauer, [arXiv:1906.06329](https://arxiv.org/abs/1906.06329).
- [47] A. Kardashin, A. Uvarov, and J. Biamonte, *Front. Phys.* **8**, 586374 (2021).
- [48] A. V. Uvarov, A. S. Kardashin, and J. D. Biamonte, *Phys. Rev. A* **102**, 012415 (2020).
- [49] M. L. Wall, M. R. Abernathy, and G. Quiroz, *Phys. Rev. Research* **3**, 023010 (2021).
- [50] P. Blagoveschensky and A. H. Phan, [arXiv:2005.14506](https://arxiv.org/abs/2005.14506).
- [51] S. Mugel, C. Kuchkovsky, E. Sanchez, S. Fernandez-Lorenzo, J. Luis-Hita, E. Lizaso, and R. Orus, [arXiv:2007.00017](https://arxiv.org/abs/2007.00017).
- [52] A. Cichocki, [arXiv:1407.3124](https://arxiv.org/abs/1407.3124).
- [53] A. Cichocki, A.-H. Phan, Q. Zhao, N. Lee, I. V. Oseledets, M. Sugiyama, and D. Mandic, *Found. Trends Mach. Learn.* **9**, 431 (2017).
- [54] I. V. Oseledets, *SIAM J. Sci. Comput.* **33**, 2295 (2011).
- [55] A. S. Bhatia, M. K. Saggi, A. Kumar, and S. Jain, *Neural Comput.* **31**, 1499 (2019).
- [56] A. Novikov, M. Trofimov, and I. Oseledets, [arXiv:1605.03795](https://arxiv.org/abs/1605.03795).
- [57] Y. LeCun, C. Cortes, and C. Burges, <http://yann.lecun.com/exdb/mnist> (AT&T Labs, Bedminster, 2010).
- [58] Y.-Y. Shi, L.-M. Duan, and G. Vidal, *Phys. Rev. A* **74**, 022320 (2006).
- [59] K. G. Wilson, *Rev. Mod. Phys.* **55**, 583 (1983).
- [60] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, [arXiv:2001.03622](https://arxiv.org/abs/2001.03622).
- [61] M. Schuld, R. Sweke, and J. J. Meyer, *Phys. Rev. A* **103**, 032430 (2021).
- [62] R. LaRose and B. Coyle, *Phys. Rev. A* **102**, 032420 (2020).
- [63] C. Roberts, A. Milsted, M. Ganahl, A. Zalcman, B. Fontaine, Y. Zou, J. Hidary, G. Vidal, and S. Leichenauer, [arXiv:1905.01330](https://arxiv.org/abs/1905.01330).
- [64] S. R. White, *Phys. Rev. Lett.* **69**, 2863 (1992).
- [65] A. Edelman, T. A. Arias, and S. T. Smith, *SIAM J. Matrix Anal. Appl.* **20**, 303 (1998).
- [66] M. Hauru, M. Van Damme, and J. Haegeman, *SciPost Phys.* **10**, 040 (2021).
- [67] I. A. Luchnikov, M. E. Krechetov, and S. N. Filippov, *New J. Phys.* **23**, 073006 (2021).
- [68] A. Altland and B. D. Simons, *Condensed Matter Field Theory* (Cambridge University Press, Cambridge, 2010).
- [69] S.-A. Cheong and C. L. Henley, *Phys. Rev. B* **79**, 212402 (2009).
- [70] W. Mnder, A. Weichselbaum, A. Holzner, J. von Delft, and C. Henley, *New J. Phys.* **12**, 075027 (2010).
- [71] M. M. Wolf, F. Verstraete, M. B. Hastings, and J. I. Cirac, *Phys. Rev. Lett.* **100**, 070502 (2008).
- [72] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning* (Esann, Bruges, Belgium, 2013), Vol. 3, p. 3.
- [73] D. C.-L. Fong and M. Saunders, *SIAM J. Sci. Comput.* **33**, 2950 (2011).
- [74] I. J. Goodfellow, J. Shlens, and C. Szegedy, [arXiv:1412.6572](https://arxiv.org/abs/1412.6572).
- [75] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, [arXiv:1810.00069](https://arxiv.org/abs/1810.00069).
- [76] S. Lu, L.-M. Duan, and D.-L. Deng, *Phys. Rev. Research* **2**, 033212 (2020).
- [77] N. Liu and P. Wittek, *Phys. Rev. A* **101**, 062331 (2020).
- [78] N. Bansal, X. Chen, and Z. Wang, Can we gain more from orthogonality regularizations in training deep CNNs? *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (2018), pp. 4266–4276.
- [79] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, *Phys. Rev. A* **99**, 032331 (2019).
- [80] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, *Phys. Rev. A* **101**, 032308 (2020).
- [81] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri *et al.*, [arXiv:1811.04968](https://arxiv.org/abs/1811.04968).
- [82] R. Sweke, F. Wilde, J. J. Meyer, M. Schuld, P. K. Fhrmann, B. Meynard-Piganeau, and J. Eisert, *Quantum* **4**, 314 (2020).
- [83] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, S. V. Isakov, P. Massey, M. Y. Niu, R. Halavati, E. Peters *et al.*, [arXiv:2003.02989](https://arxiv.org/abs/2003.02989).
- [84] J. Pino, J. Dreiling, C. Figgatt, J. Gaebler, S. Moses, M. Allman, C. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer *et al.*, *Nature (London)* **592**, 209 (2021).

- [85] J. Preskill, Lecture notes for ph219/cs219: Quantum information and computation, available at http://theory.caltech.edu/preskill/ph229/notes/chap4_01.pdf (California Institute of Technology, Pasadena, 2001), Chap. 4.
- [86] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, *Nat. Commun.* **9**, 4812 (2018).
- [87] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, *Nat. Commun.* **12**, 1791 (2021).
- [88] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, *Quantum* **3**, 214 (2019).
- [89] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *J. Mach. Learn. Res.* **12**, 2825 (2011).
- [90] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, in *Advances in Neural Information Processing Systems* 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran, Red Hook, 2019), pp. 8024–8035.
- [91] G. Vidal, *Phys. Rev. Lett.* **101**, 110501 (2008).