npj | Quantum Information

www.nature.com/npjqi

ARTICLE OPEN

Check for updates

Adaptive quantum state tomography with neural networks

Yihui Quek[1]✉, Stanislav Fort[2]✉ and Hui Khoon Ng[3,4,5]✉

Current algorithms for quantum state tomography (QST) are costly both on the experimental front, requiring measurement of many copies of the state, and on the classical computational front, needing a long time to analyze the gathered data. Here, we introduce neural adaptive quantum state tomography (NAQT), a fast, flexible machine-learning-based algorithm for QST that adapts measurements and provides orders of magnitude faster processing while retaining state-of-the-art reconstruction accuracy. As in other adaptive QST schemes, measurement adaptation makes use of the information gathered from previous measured copies of the state to perform a targeted sensing of the next copy, maximizing the information gathered from that next copy. Our NAQT approach allows for a rapid and seamless integration of measurement adaptation and statistical inference, using a neural-network replacement of the standard Bayes' update, to obtain the best estimate of the state. Our algorithm, which falls into the machine learning subfield of "meta-learning" (in effect "learning to learn" about quantum states), does not require any ansatz about the form of the state to be estimated. Despite this generality, it can be retrained within hours on a single laptop for a two-qubit situation, which suggests a feasible time-cost when extended to larger systems and potential speed-ups if provided with additional structure, such as a state ansatz.

npj Quantum Information (2021)7:105 ; https://doi.org/10.1038/s41534-021-00436-9

INTRODUCTION

Quantum state tomography (QST) is the task of estimating the density matrix of an unknown quantum state, through repeated measurements of the source, assumed to put out identical copies of the state. This procedure can be used to characterize not only quantum states, but also processes acting on quantum states, and is an indispensable subroutine in quantum information processing tasks (for example, see ref. [1] for a review). QST is, however, a resource-heavy task, as one needs many measurements on many copies of the quantum state, to yield sufficient data for a good estimate of the $d^2 - 1$ real parameter needed to describe the state of a d-dimensional quantum system. For estimating quantum processes, the number of parameters needed is d^4, a much worse scaling. Much of the research in QST is hence about minimizing the resource cost of getting an estimate with a target precision (see, among many others, refs. [2–8]).

A promising angle of attack is the use of adaptive measurements, adjusting the measurement on the next copy of the state based on the information gathered from the copies measured so far, to maximize (according to some chosen measure) the information gained from that next copy. One approach was proposed in ref. [9] (under the name *self-learning*) and generalized in ref. [10] as adaptive bayesian quantum tomography (ABQT). These methods were later experimentally implemented in refs. [11,12]. Here, the adaptation criterion, following Bayesian principles, relies on a merit function that requires an average over the posterior distribution on the state space.

For numerical tractability in handling the integration over the posterior distribution, ABQT uses a modified particle filter algorithm: The space of quantum states is discretized via samples known as *particles*, with accompanying *weights* that represent their relative probabilities according to the current prior distribution. As more data are gathered, Bayes' rule is used to update the prior distribution on the quantum state space to the posterior by updating the particle weights. Any integration over the prior or posterior distribution is then implemented as the appropriately weighted sum over the particles. The adaptive advantage of ABQT comes, however, at a high computational price. As with other filtering algorithms, a major issue is the eventual decay of the vast majority of particle weights, which undercuts the efficacy of the weight updates. In Bayesian-type tomography, this is an acute problem since the likelihood function that enters the posterior distribution quickly becomes very sharply peaked as the number of measured copies grows. Consequently, the weight becomes rapidly concentrated on a tiny subset of the particles. The numerical fix is to resample the bank periodically, i.e., the particles and weights must regularly be chosen anew. This turns out to be very computationally expensive and significantly prolongs the overall runtime of the adaptive QST algorithm.

In this paper, we demonstrate a fast, accurate, and flexible alternative to adaptive QST using a custom-built RNN architecture. We call our scheme neural adaptive quantum tomography (NAQT). Our algorithm learns from simulated experimental data to develop an efficient resampling step to refine particles, as well as a way of incorporating new measurements to its current best estimate of the state. It is also able to predict the suitable next measurement that, if performed, would lead to the most information gained given its current knowledge about the state.

The efficiency of our approach comes from using machine learning to learn an approximate replacement of the Bayesian update rule. This eliminates the problem of weight decay, and therefore, the need for time-costly resampling. All in all, our adaptive algorithm matches the performance of ABQT in terms of reconstruction accuracy, while speeding it up significantly (by a factor of up to a million, for 10^7 measurements, when run on the same computational hardware), giving a practical, genuinely on-the-fly adaptive QST

[1]Department of Applied Physics and Information Systems Laboratory, Stanford University, Stanford, CA, USA. [2]Department of Physics, Stanford University, Stanford, CA, USA. [3]Yale-NUS College, Singapore, Singapore. [4]Centre for Quantum Technologies, National University of Singapore, Singapore, Singapore. [5]MajuLab, International Joint Research Unit UMI 3654, CNRS, Université Côte d'Azur, Sorbonne Université, National University of Singapore, Nanyang Technological University, Singapore, Singapore. ✉email: yquek@stanford.edu; sfort1@stanford.edu; huikhoon.ng@yale-nus.edu.sg

scheme. The technique used to train the neural network (NN) is furthermore agnostic to the number of qubits involved and the type of measurements used and can be retrained in reasonable time to suit a particular experiment's details.

RESULTS

Neural adaptive quantum tomography

We first describe our NAQT scheme, beginning with the basic task of QST, and then explaining our approach using a NN algorithm. The technical details of our algorithm are given in the "Methods" section.

The problem: quantum state tomography. The goal of QST is to estimate the state, i.e., the density matrix, ρ, of a quantum system. ρ is a trace-1 and Hermitian operator on the d-dimensional Hilbert space $\mathcal{H}$ of the quantum system, represented by a $d \times d$ matrix. We assume that we have access to a source that puts out independent and identical copies of the (unknown) state ρ. We are allowed to make measurements on the state, and from the gathered data, estimate ρ. Generalized measurements, also known as positive operator-valued measures (POVMs), are permitted. These are describable as a set of outcome operators $\Pi \equiv \{\Pi_y\}$ on $\mathcal{H}$, satisfying $\Pi_y \geq 0 \, \forall \, y$ and $\sum_y \Pi_y = \mathbb{1}$, the d-dimensional identity operator.

For a chosen POVM Π, the probability that one gets a click in the detector for outcome Π_y is given by Born's rule, $p_y = \mathrm{Tr}(\rho \Pi_y)$. We write $\mathbf{p} \equiv \mathrm{Tr}(\rho \Pi)$ for the set of Born probabilities $\{p_y\}$. The likelihood of getting data $\mathcal{D}$, a sequence of detector clicks summarized by $\{n_y\}$, with $n_y \equiv$ number of clicks in the detector for Π_y, is

$$p(\mathcal{D}|\rho) = \prod_y (p_y)^{n_y} \ . \tag{1}$$

$p(\mathcal{D}|\rho)$ is a probability distribution over the data, i.e., $\sum_{\mathcal{D}} p(\mathcal{D}|\rho) = 1$. The ratios n_k/N, for $N \equiv \sum_k n_k$, are referred to as the relative frequencies for the measurement, and, for large N, we expect $p_k \simeq n_k/N$.

In Bayesian estimation procedures, one talks about the prior and the posterior distributions on the quantum state space. The prior distribution captures our initial knowledge about the identity of the state prior to data-taking. We denote it as $\mathrm{d}\rho \, p(\rho)$, for some suitably chosen volume measure $\mathrm{d}\rho$ on the state space. $p(\rho)$ is the prior density, while $\mathrm{d}\rho \, p(\rho)$ is the infinitesimal probability that the true state lies in the volume $\mathrm{d}\rho$, according to our prior expectations. $p(\rho)$ satisfies $\int \mathrm{d}\rho \, p(\rho) = 1$. The posterior distribution, denoted as $\mathrm{d}\rho \, p(\rho|\mathcal{D})$ represents our updated knowledge, after obtaining data $\mathcal{D}$. The update from prior to posterior densities follows from Bayes' rule

$$p(\rho|\mathcal{D}) = \frac{p(\mathcal{D}|\rho) \, p(\rho)}{p(\mathcal{D})} \ , \tag{2}$$

where $p(\mathcal{D}) \equiv \int \mathrm{d}\rho \, p(\rho) p(\mathcal{D}|\rho)$ is the likelihood of data $\mathcal{D}$.

Note that the ABQT algorithm relies on the posterior distribution to make decisions about the next measurement to make; in our NAQT scheme, as we explain below, this posterior distribution is replaced by the choices of weights on the particle samples made by the trained neural network.

Our solution: a neural network algorithm. Neural networks, and deep neural networks in particular, are a class of expressive functional approximators capable of learning complicated models across many domains, ranging from image classification[13] to game playing[14], to natural language understanding[15]. By means of gradient descent, neural networks are trained to successfully approximate an unknown function using a large number of examples and given a loss function specifying how dissatisfied one is with a solution. In cases where an exact input–output mapping is available but might be expensive to evaluate, neural networks can develop approximate, faster effective descriptions learned directly from data.

By specifying a particular neural architecture—a particular choice of neurons, their connections, and the way they interact—we describe a family of functions, with trainable parameters. Many widely used architectures encode priors on the function they are trying to approximate in their structure. This makes learning easier and typically leads to faster convergence to a solution. Such architectures are thus canonical choices for the task at hand, e.g., convolutional neural networks for image data. However, off-the-shelf network architectures are inadequate for performing tasks governed by the laws of quantum mechanics. For quantum state tomography, in particular, the adaptive mapping from previous measurement outcomes to those of the next measurement is highly nonlinear. As such, it is hard to learn simply by feeding the raw stochastic measurement outcomes into a generic, fully connected neural network.

Here, we use our knowledge of traditional tomography algorithms to construct a custom-built architecture for quantum state tomography, encoding the task explicitly into the network structure. As a crucial part of the process, it was necessary for us to develop a differentiable implementation of quantum mechanics in TensorFlow[16] that simulates POVM measurements. During the deployment of the NAQT algorithm, these are replaced by the actual measurement from the experiment.

The task for our neural network is as follows: given as input, the obtained data for a specified POVM, output a good estimate $\widehat{\rho}$ of the state ρ that gave rise to the given data. This task is encoded into our neural network structure, which bears basic structural similarities with a recurrent neural network (RNN) architecture. The recurrence comes from the fact that the network takes in outputs from previous time steps (iterations) as inputs to the unit cell at the current step, together with any new information available at the time.

While retaining the basic RNN structure, we have, however, reconfigured the unit cell for our specific purpose. A pictorial description of this is presented in Fig. 1; technical details are provided in the "Methods" section. In our setting, the recurrent nature of this architecture allows for an iteratively improving estimate of the density matrix with every new batch of measurement data. In particular, it can be stopped after any number of measurements, and since (as we will see) on average, the distance to the true solution decreases, it provides the best estimate up to that moment. It is also guaranteed to output a valid density matrix as its running estimate of the state is always a convex linear combination of the current bank of particles, each a valid state.

Our algorithm shares structural similarities with ABQT. Both employ a bank $\mathcal{B}$ of particles $\{\rho_i\}$—candidate quantum states—and associated weights $\{w_i\}$; we write $\mathcal{B} = \{(\rho_i, w_i)\}$. At any step in the algorithm, $\mathcal{B}$ encapsulates the current knowledge of the state given the data gathered so far. All averages over the state space required in both algorithms are computed using $\mathcal{B}$

$$\langle f(\rho) \rangle \equiv \sum_{i \in \mathcal{B}} w_i f(\rho_i). \tag{3}$$

In particular, the average state over $\mathcal{B}$, $\widehat{\rho} \equiv \langle \rho \rangle$ is the current best estimate of the true state. There is a crucial difference, however, between the bank for ABQT and that for NAQT: in ABQT, $\mathcal{B}$ is a discrete approximation of the posterior distribution from the data gathered so far; in NAQT, $\mathcal{B}$ is not constructed from the usual Bayesian update rule, but is chosen by the neural network based on its past training.

After every new round of measurements, both algorithms use the new data to take a bank-update step as well as a measurement

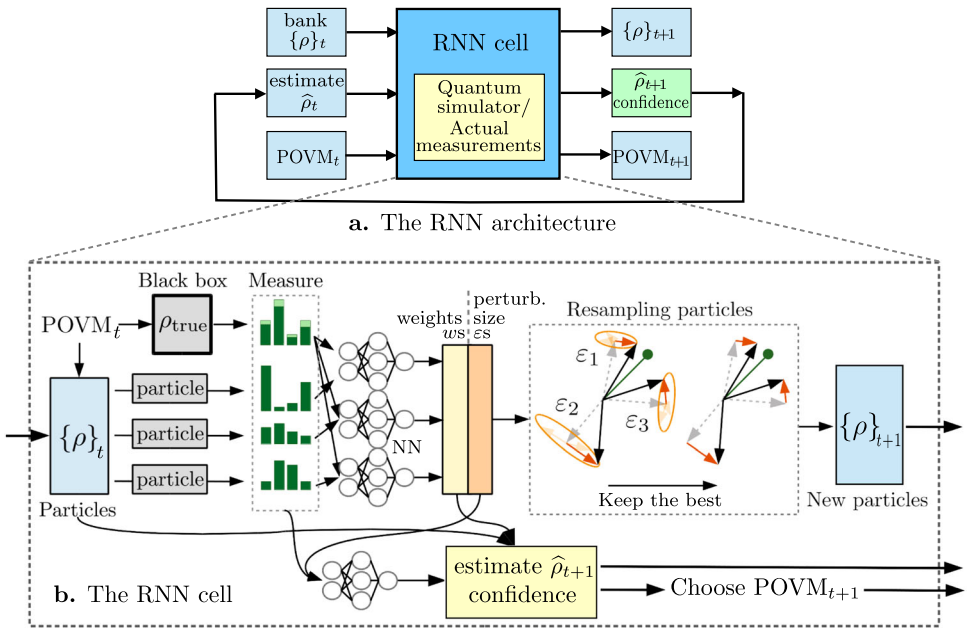

Fig. 1 Our custom-built recurrent neural network (RNN) architecture. After every round of measurement, generating a new set of data, the RNN cell takes a previously generated bank of weighted candidate states—"particles"—and the best guess for the (unknown) true state as inputs and outputs a refined bank of particles together with a new estimate of the true state and an associated confidence value. In addition, the refined bank is used to select the next measurement to be performed. This process is iterated, gradually converging to the correct density matrix. During training, there are two stages of analysis for each measurement round: the first stage updates the weights in the bank; the second stage updates the particles.

adaptation step. In the bank-update step, the particles and/or weights in the bank are modified based on the new data. In ABQT, only the weights are changed in each round, to update the bank from representing the prior (before the new data) to the posterior (after the new data) according to Bayes' rule; the particles in the bank are changed only when a resampling of the posterior due to the weight decay problem is deemed necessary. In NAQT, however, the weights and particles are perturbed every round, as decided by the neural network, to (hopefully) move $\widehat{\rho}$ closer to the true state.

In the measurement adaptation step, both ABQT and NAQT employ the same information-gain quantity proposed in ref. [10]

$$\mathcal{I}(\Pi,\mathcal{D}) \equiv H(\mathbf{p}(\Pi|\mathcal{D})) - \langle H(\mathbf{p}(\Pi|\mathcal{D}))\rangle. \tag{4}$$

Here, $\mathcal{D}$ are the data gathered so far, and Π is the POVM to be measured on the next copy of the state. $\mathbf{p}(\Pi|\mathcal{D})$ denotes the expected set of Born probabilities for the next measurement, given $\mathcal{D}$: $\mathbf{p}(\Pi|\mathcal{D}) \equiv \{\langle \mathrm{Tr}(\Pi_y\rho)\rangle\}_y$ [recall Eq. (3)]. $H(\{p_y\}) \equiv -\sum_y p_y \log p_y$ is the Shannon entropy for a set of probabilities $\{p_y\}$. $\mathcal{I}(\Pi,\mathcal{D})$ is hence the entropy (the first term in $\mathcal{I}$) of the outcomes from measuring Π on the next copy, predicted using the information accumulated from $\mathcal{D}$, less the inherent entropy (the second term, which is averaged over the current bank of states) for that measurement given our current state of knowledge. The measurement adaptation is done by choosing the Π, from some pre-determined set $\mathcal{P}$ of POVMs (e.g., those experimentally accessible), that maximizes $\mathcal{I}(\Pi,\mathcal{D})$. The intuition of maximizing the entropy stems from the desire to measure the next copy in a "direction" of maximal uncertainty—or minimal information—given the data observed so far.

The ability of NAQT to give a reliable estimate of the true state through updating the particle bank round by round comes from the training of its neural network. In the training phase, the neural network learns the best bank-update rules from the data simulated from training examples. Details of this training phase are given in the "Methods" section. We note here that, in contrast to ABQT where the update of the particles in the bank is a resampling step done only when needed to fix the practical problem of weight decay over multiple rounds, in NAQT, the particle update is a crucial inferential step that, together with the weight update, moves the candidate particles toward the true state. This is reminiscent of the narrowing of the Bayesian posterior, as in ABQT toward the true state as more data are collected.

Application: two-qubit tomography

To illustrate the strength of our NAQT scheme, we apply it to two-qubit state tomography, the case investigated in ref. [12]. We benchmark the performance of NAQT against ABQT. We implemented all tests in the Python 3 programming language, for proper runtime comparisons with NAQT written in TensorFlow and Python, and verified our implementation of ABQT against the results in ref. [12]. For each tomography scheme, the accuracy of the state reconstruction is evaluated using the squared Bures distance between the true state ρ and the estimate $\widehat{\rho}$

$$d_{\mathrm{B}}^2(\rho,\widehat{\rho}) \equiv 2[1 - F(\rho,\widehat{\rho})]. \tag{5}$$

Here, $F(\rho,\sigma) \equiv \mathrm{Tr}\left(\sqrt{\rho^{1/2}\sigma\rho^{1/2}}\right)$ is the fidelity between two states ρ and σ. Following ref. [12], we consider only *product* POVMs, $\Pi \equiv \Pi^{(1)} \otimes \Pi^{(2)} \equiv \{\Pi_{y_1}^{(1)} \otimes \Pi_{y_2}^{(2)}\}$, where each $\Pi^{(i)}$ is a one-qubit POVM for qubit i, and y_i labels the individual outcomes. Product POVMs form the class of multiqubit measurements most easily accessible in experiments.

We compare the performance of NAQT to ABQT for the projective product POVMs carried out in ref. [12], i.e., we measure a (two-outcome) basis for each qubit; we refer to such POVMs as *basis POVMs*. The degrees of freedom to be optimized in the adaptive procedure correspond to the choice of bases to be measured, one for each qubit. Figure 2 shows this comparison. In our simulations, NAQT uses a particle bank with 100 particles; we run ABQT with different particle bank sizes to study the

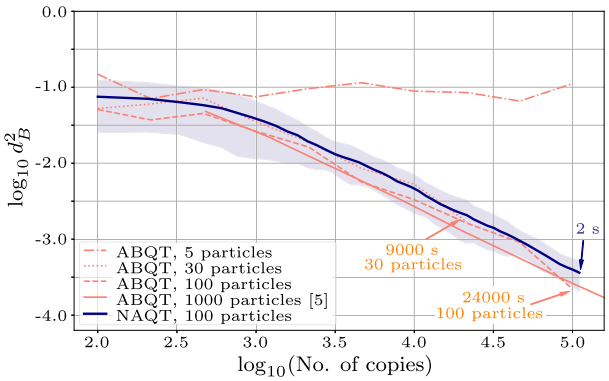

Fig. 2 Performance comparison for basis POVMs. Our neural network algorithm NAQT with 100 particles is compared against ABQT runs with 5, 30, and 100 particles; the data for ABQT with 1000 particles are extracted from ref. [12]. The plot gives the reconstruction accuracy against a number of copies measured. The light-blue shading gives the 1σ error bars for NAQT. Our NAQT algorithm performs comparably, in terms of reconstruction accuracy, to ABQT; the runtimes are, however, significantly—by a factor of a few thousand—more favorable for NAQT than ABQT.

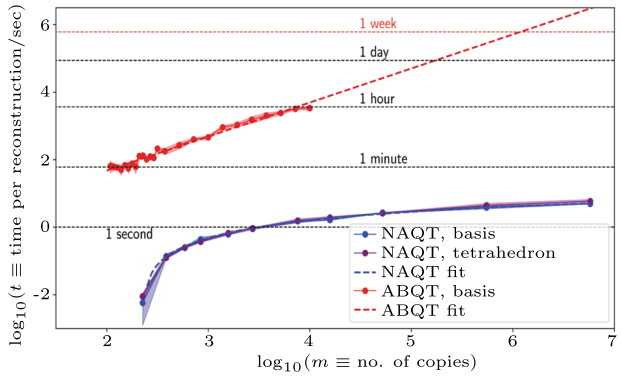

Fig. 3 Scaling of reconstruction runtime with the number of copies measured, using the basis POVM, as well as the product tetrahedron POVM (for NAQT; see main text), all with 100 particles. The runtime curves are fitted to the following lines: for NAQT, $t = -3.5\text{s} + (1.3\text{s})\log_{10}(m + 300)$; for ABQT, $t = (0.4\text{s})m$. Comparisons were run on the same computational hardware. For $>10^5$ copies, we had to extrapolate the ABQT runtimes, as each reconstruction was taking tens of hours by then. The NAQT runtime t is observed to scale logarithmically with m, the number of copies measured, while ABQT appears to scale linearly. For 10^7 measurements, our algorithm completes a reconstruction in ≈5 s, while the ABQT would take more than a week (extrapolated).

dependence of runtime and reconstruction accuracy on the particle number. Generally, ABQT performs better as more particles are used, since better estimates can be made when the posterior distribution—crucial to the ABQT heuristic—is more finely sampled. In our experiments, the performance of NAQT improves with more particles as well, but for computational reasons (limited memory on a laptop), we could not efficiently explore more than 100 particles for NAQT. ABQT's performance anyway seems empirically to the plateau beyond 30 particles, so the comparison with 100 particles suffices to illustrate our points here.

With 100 particles, the performance of NAQT is comparable in reconstruction accuracy to ABQT. However, observe the significantly shorter runtime needed for NAQT to achieve that accuracy: 2 s for NAQT for a total of 100,000 copies measured, compared with 24,000 s (≃6.7 h) for ABQT. This runtime improvement is further illustrated in Fig. 3. There, we plot the scaling of computational runtimes for the two QST schemes with the number of copies measured. ABQT takes prohibitively long runtimes that appear to be polynomial (approximately linear, according to our experiments) in the number of copies measured. In contrast, the runtime of NAQT seems to be logarithmic in the number of copies. For 10^7 copies measured, our NAQT converges to a solution in approximately 5 s on a laptop, while ABQT would probably take more than a week to complete.

For a broader comparison, we also plot the runtimes for NAQT with a product tetrahedron POVM, instead of the basis POVM. The tetrahedron POVM is the symmetric informationally complete POVM for a single qubit, so named because the POVM outcomes form a regular tetrahedron in the Bloch ball[17]. Here, the measurement adaptation involves choosing the orientation of the tetrahedron relative to a reference direction. Our numerical experiments with the product tetrahedron POVM show a nearly identical runtime behavior as for the basis POVM, suggesting the robustness of our NAQT runtime speedup for different POVM choices.

Why is this runtime speedup so dramatic? In ABQT, the resampling step is triggered whenever the posterior—determined by the Bayes' update rule given the data—becomes too sharply peaked about too few particles in the bank ("peakedness" can be measured, for example, by the effective sample size criterion $\left[\sum_s w_s^2\right]^{-1}$ of ref. [12]). In NAQT, the neural network updates allow us to avoid the weight decay problem encountered in ABQT. For us, when resampling occurs is not dictated by how the posterior distribution evolves according to Bayes' rule. Instead, we have free rein to choose our own schedule for resampling the particles. The neural network, in the training phase, learns the best weight-update rule given the specified resampling schedule. In our examples, we chose the resampling to take place at intervals of exponentially increasing size, that is t_1, $2t_1$, $4t_1$, etc., for some initial t_1 value. The number of resampling events thus scales only logarithmically with the total number of copies measured. Since the resampling events are the rate-determining step, the runtime inherits this logarithmic scaling, as observed in Fig. 3.

DISCUSSION

Our NAQT scheme performs comparably to ABQT in reconstruction accuracy but shows an orders-of-magnitude reduction in computational time. The "secret sauce" here is the incredible expressive power of neural networks as functional approximators; our resampling step uses them to learn an effective heuristic for state perturbations directly from data, sidestepping ABQT's computationally expensive resampling step.

It is worthwhile to note the ease in accommodating a large variety of measurement choices within our NAQT algorithm. The neural network has to be re-trained every time, of course, but the only change in the code needed is the specification of the chosen POVM, so that the neural network understands the relationship between the measurement data and the state, effectively learning Born's rule. Gone is the need, encountered in many tomography schemes, for the invention of novel estimators or ad hoc fixes because of different expressions of the positivity constraints for different POVMs. The neural network generates only physically valid particles, and the estimator is immediately generated from the neural network-updated particle bank. There is also no need for an initial ansatz for the quantum state, as needed in some alternate approaches (see, for example, ref. [18]).

Our work thus shows that neural networks can adaptively estimate an arbitrary density matrix, in a fast, accurate, and flexible manner. In contrast to related approaches that directly learn the description of the quantum state from given measurement data,

our algorithm has an additional layer of complexity: it learns the optimal measurements to learn the state.

One key feature of neural networks is their versatility, and we can think of any further applications of our algorithm with little added effort. For example, one can easily adapt our neural network framework to estimate, not general states, but states coming from a specific class, e.g., pure or low-rank states[19,20], the Choi matrix for a particular category of quantum channels, etc. One only needs to restrict the training examples and the particle update to only states within that class. It is equally straightforward to retrain our algorithm to return not the full density-matrix description of the state, but functions of the state, such as entanglement entropy, fidelity to a target state (see ref. [21] for the standard nonneural-network procedure for this), etc. One could even imagine extending our algorithm to deal with noisy measurements: Simply simulate the noise in the training phase.

In building our RNN framework, we benefitted from the existence of a streamlined computational framework for machine learning, which enabled us to train resampling and adaptive components of our algorithm jointly using the same loss function, such that the output of one was calibrated optimally for the other. Interestingly, Kim et al.[22], in a work similar in spirit to ours but in a different subject domain, report similar benefits from the end-to-end concatenation of their neural network-based encoder and decoder for the design of sequential codes for channel communication.

A hurdle to QST of high-dimensional states is the fact that computational complexity grows exponentially in the number of qubits to be estimated. This is unavoidable in any tomography algorithm since the dimension of the required output itself grows exponentially. However, neural networks, and machine learning more broadly, could be the key to achieving a favorable pre-factor for this exponential complexity, which would push the current tractable dimension boundary even higher. Our work shows a very significant runtime speedup for the simplest two-qubit example, but the same principles that led to the speedup—that of replacing Bayes' rule updating by trained neural network updates on a planned resampling schedule—are not specific to two qubits. We thus expect the advantage to hold even in higher dimensions.

METHODS

A SINGLE TRAINING EXAMPLE

Here, we describe the main component underlying our work—the training of the neural network (NN) for NAQT. We organize our description as pseudocode that closely follows the actual computer code for a single training example, i.e., one choice of the true quantum state ρ_{true}. The training can be done in the adaptive or random (i.e., no measurement adaptation) mode.

Algorithm 1. Training example

Require: True state ρ_{true}; randomly generated bank $\mathcal{B} \equiv \{(\rho_i, w_i)\}_{i=1}^{N_{\text{bank}}}$; number of copies measured in each time step $\{M_t\}_{t=1}^{T}$; set of times at which we resample $\mathcal{S}_r$; set of times at which we adapt the POVM $\mathcal{S}_a$. ▷ Note A.

Ensure: Sequence of reconstructions $\{\widehat{\rho}_t\}_{t=1}^{T}$ and associated losses. Total reconstruction loss for training use.

T rounds of measurements
for time step t from 1 to T **do**
if $t = 1$ or in random mode **then**
Generate a random POVM Π.
else ▷ in adaptive mode
Use POVM Π from previous time step.
end if
Data from ρ_{true} ▷ NN has no direct access to ρ_{true}.
Simulate data $\mathcal{D}_t \equiv \{n_y\}$ for measuring Π on M_t copies of ρ_{true} using Born's rule. Set $\widehat{\mathbf{p}} = \{n_y/M_t\}_y$ as the empirical Born probabilities for ρ_{true}.
Update the weights
for member ρ_i of particle bank $\mathcal{B}$ **do**
Calculate Born probabilities $\mathbf{p}_i = \text{Tr}(\Pi\rho_i)$;
Calculate L_1 and L_2 distances between $\mathbf{p}_i$ and $\widehat{\mathbf{p}}$.
NN(distances) → combined distance ▷ note (B)
NN(distances) → ε_i ▷ perturbation size
NN(distances) → w_i ▷ new weights
end for
$\sum_i \rho_i w_i \to \rho_{\text{guess}}$ ▷ Best guess from this step
NN(combined distance, weights) → guess the score
if t in $\mathcal{S}_r$ **then**
Update the particles
Draw N_{bank} particles $\{\rho_i'\}$ from $\mathcal{B}$ at random according to weights $\{w_i\}$.
ρ_i' inherits the ε value $(\equiv \varepsilon_i')$ for the particle it corresponds to in the original bank $\mathcal{B}$.
for step s from 1 to N_{steps} **do** ▷ note (C)
for member of new particle bank ρ_i' **do**
Purify $\rho_i' \to \vec{v}_i$. ▷ note (D)
Generate N_{resample} random vectors $\{\vec{u}\}$.
for random vector $\vec{u}$ **do**
Keep the part $\vec{o}$ of $\vec{u}$ orthogonal to $\vec{v}_i$.
Normalize $\vec{o}/|\vec{o}| \to \vec{o}$.
Let $(\vec{v}_i)^{\text{perturb}} \equiv (1-\varepsilon_i')\vec{v}_i + \varepsilon_i'\vec{o} \to (\rho_i')^{\text{perturb}}$.
Calculate $(\mathbf{p}_i')^{\text{perturb}} \equiv \text{Tr}(\Pi(\rho_i')^{\text{perturb}})$.
Calculate distance between $(\mathbf{p}_i')^{\text{perturb}}$ & $\widehat{\mathbf{p}}$.
end for
$(\rho_i')^{\text{perturb}}$ with smallest distance → new ρ_i.
end for
New bank of particles → $\mathcal{B}$
end for
end if
Current best estimate and the loss
From step t we have ρ_{guess}, guess score, and $\mathcal{B}$.
Using guesses from steps $< t$, calculate guessweights = softmax(guessscores). ▷ note (E)
$\widehat{\rho}_t = \sum_{t' \le t} \text{guessweight}_{t'}\ \rho_{\text{guess},t'}$ ▷ output guess
$\text{Loss}_t = \text{Distance}(\widehat{\rho}_t, \rho_{\text{true}})$
Adaptation of POVM
if in adaptive mode and t in $\mathcal{S}_a$ **then**
Generate a set $\boldsymbol{\Theta} = \{(X_i, Y_i, Z_i)\}_{i=1}^{N_{\text{sample}}}$ of random rotation angles about the X, Y, Z axis.
for i in N_{sample} **do**
Consider POVM parametrized by $\boldsymbol{\theta}_i \to \Pi(\boldsymbol{\theta}_i)$.
Calculate Born probabilities p_{mean} of measuring the POVM $\Pi(\boldsymbol{\theta}_i)$ on the output guess $\widehat{\rho}_t$
For all members of the particle bank i obtain outcome probabilities $\mathbf{p}_i$ of measuring the POVM on the particle ρ_i.
Calculate a mixed entropy heuristic $\mathcal{I} = S(p_{\text{mean}}) - \sum_i w_i S(p_i)$, where $S(p)$ is the entropy of distribution p.
end for
Choose $\Pi(\boldsymbol{\theta})$ with the largest $\mathcal{I}$ value → Π.
end if
end for
$\text{Loss} = \sum_{t=1}^{T} \text{Loss}_t$ ▷ note (F)

A few notes referenced in the pseudocode:

(A) $\mathcal{S}_r$, a subset of the times $t = 1, 2, \ldots, T$, specifies the resampling schedule; $\mathcal{S}_a$ is a subset of times when the POVM is adapted. In our examples, $\mathcal{S}_r$ and $\mathcal{S}_a$ are the same as the set of times $\{t\}$, i.e., every time a resampling happens,

a POVM adaptation is also done, and an interim estimate of the state is put out. $T = 50$ in our examples. The number of measurements at each time step is exponentially increasing, i.e., they occur in batches of size $b = 100 \times 1.206t$, for $t = 0, 1, \ldots, 49$, making for a total of 50 batches of increasing sizes $120, 145, \ldots 10^6$. One could, in general, adjust these choices to see if even better performance can be obtained.

(B) The $L_1 (\equiv \sum_i |\mathbf{p}_i - \widehat{\mathbf{p}}_i|)$ and $L_2 (\equiv \sum_i (\mathbf{p}_i - \widehat{\mathbf{p}}_i)^2)$ distances between $\mathbf{p}_i$ and $\widehat{\mathbf{p}}$ are computed and then given to the NN. The NN is allowed to decide how it wants to combine the two distances into a single distance measure ("combined distance" in the pseudocode) by adjusting the relative weights of the L_1 and L_2 distances. This gives flexibility in how the deviation of $\mathbf{p}_i$ from $\widehat{\mathbf{p}}$ is quantified; if one distance works better than the other, the neural network discovers this in the process of the training. More distance measures can be given to the NN, to gain even greater flexibility, if desired.

(C) This for-loop does a Metropolis–Hastings Monte-Carlo-like update (following ref. [12]) of the set of particles, moving the particles by N_{steps} small steps, each time in some best direction chosen from among a set of possibilities $\{\overrightarrow{u}\}$.

(D) Following[12] (see Appendix C of that paper), we represent an n-qubit density matrix ρ as a pure state, denoted as $\overrightarrow{v}$, in the extended Hilbert space of dimension 2^{2n}.

(E) The guess scores quantify the algorithm's confidence in its guess for the unknown state at time t. The softmax function is a standard way in machine learning to normalize the guess scores so that they sum to 1.

(F) The loss output in this step is the loss on a single training example, which comprises a single run that measures all the copies (i.e., iterates through t) of a single state. Throughout such a run, even though the algorithm is adaptively choosing the POVM orientation, the parameters of the intermediate neural networks used (in, for example, the weight-update step) may not be optimal. The loss is thus fed into an outer training loop that adjusts these parameters on the next training example.

The above is used in the training phase of NAQT. In the actual use of the trained NN, the procedure is exactly as described above for a single training example, except that, rather than $\widehat{\mathbf{p}}$ computed from the simulated data from the true state, it is given by the actual experimental data obtained from measuring the POVM on the state source.

DATA AVAILABILITY

The data generated for the two-qubit tomography example are available from the authors on reasonable request.

CODE AVAILABILITY

The code for the RNN architecture for the two-qubit tomography example is available from the authors on reasonable request.

Received: 29 October 2020; Accepted: 2 May 2021;
Published online: 24 June 2021

REFERENCES

1. Paris, M. & Řeháček, J. *Quantum State Estimation*, vol. 649 of *Lecture Notes in Physics* (Springer-Verlag, Berlin, Heidelberg, 2004).
2. Aaronson, S. The learnability of quantum states. *Proc. Math. Phys. Eng. Sci.* **463**, 3089–3114 (2007).
3. Cramer, M. et al. Efficient quantum state tomography. *Nat. Commun.* **1**, 149 (2010).
4. Senko, C. et al. Coherent imaging spectroscopy of a quantum many-body spin system. *Science* **345**, 430–433 (2014).
5. Wiebe, N., Granade, C., Ferrie, C. & Cory, D. G. Hamiltonian learning and certification using quantum resources. *Phys. Rev. Lett.* **112**, 190501 (2014).
6. Haah, J., Harrow, A. W., Ji, Z., Wu, X. & Yu, N. Sample-optimal tomography of quantum states. *IEEE Trans. Inf. Theory* **63**, 5628–5641 (2017).
7. Aaronson, S., Chen, X., Hazan, E., Kale, S. & Nayak, A. Online learning of quantum states. *J. Stat. Mech.* **2019**, 124019 (2019).
8. Rocchetto, A. et al. Experimental learning of quantum states. *Sci. Adv.* **5**, eaau1946 (2019).
9. Fischer, D. G., Kienle, S. H. & Freyberger, M. Quantum-state estimation by self-learning measurements. *Phys. Rev. A* **61**, 032306 (2000).
10. Huszár, F. & Houlsby, N. M. T. Adaptive bayesian quantum tomography. *Phys. Rev. A* **85**, 052120 (2012).
11. Hannemann, T. et al. Self-learning estimation of quantum states. *Phys. Rev. A* **65**, 050303 (2002).
12. Struchalin, G. I. et al. Experimental adaptive quantum tomography of two-qubit states. *Phys. Rev. A* **93**, 012103 (2016).
13. LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **86**, 2278–2324 (1998).
14. Silver, D. et al. Mastering the game of go without human knowledge. *Nature* **550**, 354– (2017).
15. Wu, Y. et al. Google's neural machine translation system: bridging the gap between human and machine translation. *CoRR* Preprint at *arXiv* https://arxiv.org/abs/1609.08144 (2016).
16. Abadi, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. https://www.tensorflow.org/ (2015).
17. Řeháček, J., Englert, B.-G. & Kaszlikowski, D. Minimal qubit tomography. *Phys. Rev. A* **70**, 052321 (2004).
18. Torlai, G. et al. Neural-network quantum state tomography. *Nat. Phys.* **14**, 447–450 (2018).
19. Gross, D., Liu, Y.-K., Flammia, S. T., Becker, S. & Eisert, J. Quantum state tomography via compressed sensing. *Phys. Rev. Lett.* **105**, 150401 (2010).
20. Flammia, S. T., Gross, D., Liu, Y.-K. & Eisert, J. Quantum tomography via compressed sensing: error bounds, sample complexity and efficient estimators. *New J. Phys.* **14**, 095022 (2012).
21. Flammia, S. T. & Liu, Y.-K. Direct fidelity estimation from few pauli measurements. *Phys. Rev. Lett.* **106**, 230501 (2011).
22. Kim, H. et al. Communication algorithms via deep learning. in *International Conference on Learning Representations*. https://openreview.net/forum?id=ryazCMbR (2018).

ACKNOWLEDGEMENTS

S.F. was supported by a Research Assistantship at the Kavli Institute for Particle Astrophysics and Cosmology at Stanford University. Y.Q. was supported by a Stanford Graduate Fellowship and a National University of Singapore Overseas Graduate Scholarship. This work is funded by the Singapore Ministry of Education (through the Academic Research Fund Tier 2 MOE2016-T2-1-130) and the National Research Foundation of Singapore.

AUTHOR CONTRIBUTIONS

H.K.N. conceived the original idea and supervised the work. Y.Q. and S.F. devised and implemented the RNN architecture, and carried out the numerical simulations, sharing the work equally between them. All authors contributed to the discussion of the results and the writing of the paper.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Correspondence and requests for materials should be addressed to Y.Q., S.F. or H.K.N.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.