Modern day quantum simulators can prepare a wide variety of quantum states but extracting observables from the resulting "quantum data" often poses a challenge. We tackle this problem by developing a quantum state tomography scheme which relies on approximating the probability distribution over the outcomes of an informationally complete measurement in a variational manifold represented by a convolutional neural network. We show an excellent representability of prototypical ground- and steady states with this ansatz using a number of variational parameters that scales polynomially in system size. This compressed representation allows us to reconstruct states with high classical fidelities outperforming standard methods such as maximum likelihood estimation.

Furthermore, it achieves a reduction of the root mean square errors of observables by up to an order of magnitude compared to their direct estimation from experimental data.

A new class of variational ansatz functions, which has been employed for QST recently are neural network quantum states (NQS) [14]. The exploration of NQS is motivated by universal approximation theorems [17] and the observation that many NQS can efficiently encode volume-law entanglement and thereby have higher representational power compared to most tensor-network based approaches [19] as well as a favorable generalization to higher dimensions. These emerging neural network QST (NN-QST) approaches [15–16] are starting to receive attention from the experimental communities with applications to Rydberg-, trapped-ion and optical systems [27–39].

A major challenge for the successful application of NN-QST schemes is the choice of the variational ansatz, i.e. the network architecture, and the understanding of its intrinsic limitations. Recently, it was shown that convolutional neural networks (CNNs) are capable of efficiently encoding volume-law entanglement [21] for pure states, motivating us to explore this architecture also for NN-QST tasks. We use CNNs to learn a state's probabilistic representation from measured data, bridging the gap between quantum theory and neural networks by employing Positive Operator Valued Measures (POVMs).
resulting QST scheme scales subexponentially in the system size and makes no assumption on the target state’s purity, generalizing previous approaches [15, 16]. In order to understand in which regimes one may expect to profit from employing this NN-QST, we perform extensive comparisons to traditional methods, providing a so-far missing benchmarking approach, and also point out potential pitfalls inherent to NN-QST. For a broad range of typical experimental scenarios, we see improvements compared to MLE for small datasets, as well as a noise reduction for the estimation of local observables on larger systems, thus decreasing the necessary amount of experimental samples at a given error threshold.

POVM formalism. For efficient NN-QST a suitable variational ansatz is crucial. Here we adopt an approach pioneered in [21], where the quantum state is encoded by a probability distribution over the outcomes of experimental measurements. This probability distribution, in turn, is approximated by a neural network. This has the advantage that it allows for a tomography scheme that is directly compatible with experimentally measured data and allows for applying standard probabilistic machine learning models operating on real numbers. In the following we briefly summarize the employed learning rule and the POVM formalism, and direct to [21], as well as [29, 32, 33] for further details on NQS in combination with POVMs.

In the probabilistic formulation of quantum mechanics, the state of a system is not represented by its density matrix $\rho$, but by the Born-rule probabilities of an informationally complete (IC) POVM, consisting of measurement operators $M_a$

$$P(a) = \text{Tr}[\rho M_a].$$  \hfill (1)

‘Informationally complete’ means that the $M_a$ form a complete basis for the set of hermitian operators, implying that any density matrix or observable can be expanded using the POVM operators. Therefore, knowing $P(a)$ is equivalent to knowing $\rho$ and in principle $\rho$ can be inferred from $P(a)$ by inverting Eq. (1) to $\rho = P(a)T_{an}^{-1}M_a$, using the overlap matrix $T_{an} = \text{Tr}[M_aM_a^\dagger]$ and by summing over repeated indices. Any observable may be computed by sampling from the POVM distribution using

$$\text{Tr}[\rho O] = \sum_a P(a)O_a = \langle O_a \rangle_{a \sim P(a)},$$  \hfill (2)

where $O_a = \text{Tr}[M_aO]T_{an}^{-1}$ is the "POVM representation" of an observable $O$. We consider systems of $N$ qubits where each qubit is read out individually and thus the POVM elements are product operators $M_a = M_{a_1} \otimes \ldots \otimes M_{a_N}$. With such a factorized POVM, the overlap matrix $T$ also factorizes, which allows the computation of $O_a$ to be efficient, given that $O$ is a local observable.

The experimentally most convenient IC measurement scheme consists of single qubit Pauli measurements. By randomly selecting the $x$, $y$ or $z$ basis for measuring each qubit in each experimental run, one effectively measures a POVM with 6 possible outcomes per qubit. Unfortunately, this results in an overcomplete POVM, with an overlap matrix $T$ that is not invertible. This can easily be remedied by grouping three out of the six outcomes into one, resulting in e.g.

$$\{M_{0/1/2} = \frac{1}{3} \left| \langle t_{x/y/z}^z \rangle \langle t_{x/y/z} \rangle \right|, M_3 = 1 - M_0 - M_1 - M_2\}.$$  

This POVM is typically referred to as the Pauli-4 POVM [21]. In this setting, a measurement outcome $a = a_1a_2\ldots a_N$ is a string of $N$ single qubit outcomes for an $N$ qubit system, over which the POVM distribution $P(a) = P(a_1, a_2, \ldots, a_N)$ is defined. A dataset of size $N_s$ is a set of multiple such outcomes $D = \{a_1, a_2, \ldots, a_{N_s}\}$.

We use a neural network as a non-linear function that returns a single POVM probability $P(a)$ given a POVM outcome $a$ as input. We can thus write our neural network ansatz for the POVM distribution as $P_{NN}(a)$, where $\theta$ denotes the tuple of variational parameters describing the ansatz. Fitting the variational parameters of the network builds on standard machine learning methods: We use the established ADAM [34] optimizer to find the
parameters $\theta$ that have the highest likelihood of reproducing the data. This is equivalent to minimizing the cross-entropy between the dataset distribution and the NN distribution.

We finally point out, that the positivity condition on the density matrix is not known to easily translate into the POVM formalism, without introducing exponential scaling \[35\]. Therefore, one cannot guarantee that all sampled observables obey physical constraints (such as $\text{Tr}[\rho^2] < 1$). Typically, however, the negativity of the density matrix is not the limiting factor for the observables we are interested in.

**CNN architectures.** NN-QST in conjunction with POVMs has been successfully applied to recurrent neural networks (RNNs) \[21\] as well as attention based models \[23\]. We expand on this knowledge, by employing two distinctly different versions of CNNs, motivated by recent theoretical developments, which frame CNNs as a generalization to matrix product states \[19,31\].

The first architecture we consider is the "standard" CNN in Fig. 1 (upper left), which is fed a one-hot encoded vector of single-site outcomes, performs repeated convolutions on it and returns the corresponding (unnormalized) POVM probability $P_{\text{NN}}(a)$. Motivations for using this architecture are its established representational power when operating on pure states \[19,31\], the ability to represent 1D and 2D states, as well as the ability to encode certain symmetries such as translation invariance without increasing the computational complexity. Unfortunately, one has to resort to Markov Chains in order to draw samples from the network, which can lead to unintended correlations between generated samples.

The second architecture is a 1D autoregressive CNN (ARCNN) \[36,37\], illustrated in Fig. 1 (upper right). This architecture is very similar to the standard CNN, but makes use of the autoregressive property, which states that a probability of multiple variables can be partitioned into a product of conditionals: $P(a) = P(a_1)P(a_2|a_1)\ldots P(a_N|a_1,\ldots,a_{N-1})$. Other network architectures that also make use of this property include recurrent neural networks \[21\] and Transformers \[23\]. The autoregressive property of the ARCNN allows probabilities to be exactly normalized, and for samples to be drawn exactly, circumventing the need for Markov Chains. More details on both architectures can be found in the appendix.

**Expressivity of CNNs.** For both the CNN with product-output layer (see appendix) and the ARCNN, one can write down a maximum (physical) distance $d_{\text{max}}$, beyond which correlations are typically not captured correctly anymore:

\[
\begin{align*}
    d_{\text{max}}^{\text{CNN}} &= (K-1)L, \\
    d_{\text{max}}^{\text{ARCNN}} &= (K-1)L + 1,
\end{align*}
\]

where $L$ is the network depth and $K$ the size of the convolution kernels. Both can be understood intuitively from the schematics in Fig. 1 (upper): With each layer, the dependence of any single site outcome gets propagated by a distance of one less than the kernel size. For the autoregressive CNN, an input of $a_i$ results in the probability for $a_{i+1}$, explaining the fact, that correlations reach one site further as compared to the CNN. These cutoff points for correlations can be seen in Fig. 1 (lower).

For the following results, we are typically interested in systems with long-range correlations, hence we use architectures with $d_{\text{max}} = l$, where $l$ is the (side-) length of (2D) 1D systems. This gives a very compact state representation, with the number of variational parameters scaling only cubically in the system size. We typically do not find a benefit in increasing $d_{\text{max}}$ beyond this point, as this increases the number of network parameters, artificially enlarging the search space and increasing the risk of overfitting, resulting in worse generalization. In the following it will be crucial to quantify how well the networks generalize, i.e. how many samples one needs to learn a state well, or equivalently, what errors one can expect for given dataset sizes.

**NN-QST benchmarks.** We benchmark the tomography scheme by comparing it to two conventional schemes: MLE, as it presents the go-to choice for many small scale qubit systems \[8,38\], as well as direct estimation of observables from the dataset. To carry out this comparison we generate synthetic measurement data sets. We compute a target density matrix exactly and compute its POVM distribution, from which we draw samples (typically $1k - 10k$ for 16 qubit systems). We

\[
\begin{align*}
    N_s &= 0.05 \cdot 4^k, \\
    N_s &= 0.5 \cdot 4^k, \\
    N_s &= 2 \cdot 4^k,
\end{align*}
\]
then use these samples to train the network, resulting in the optimal variational representation $P_{\text{NN}}(a)$. For small systems, i.e. those where it is feasible, we perform MLE to obtain a MLE-estimate for the density matrix, of which we compute the POVM distribution $P_{\text{MLE}}(a)$. We then ask, which of these two estimates is closer to the ground truth target distribution, as measured by the classical infidelity

$$D_{\text{NN/MLE}} = 1 - \sum_a \sqrt{P_{\text{NN/MLE}}(a)P_{\text{Truth}}(a)}.$$  

(5)

This allows us to quantify, which of the two estimates is better, by using the quotient $D_{\text{NN}}/D_{\text{MLE}}$. If it is less than one, the network gives the better estimate for the target state compared to MLE, and vice versa.

For systems where MLE is infeasible, we instead consider the root mean square (RMS) error of local observables

$$\text{RMS}_{\text{NN/Data}} = \sqrt{\langle (O_{\text{NN/Data}} - O_{\text{Truth}})^2 \rangle},$$

(6)

inspired by [22]. These can either be computed from the training dataset itself, or from the network-encoded distribution, from which we typically draw 500k samples for 16 qubit systems. In this situation the NN acts in a way of replacing the measured dataset with a larger, network-generated one, aiming to decrease statistical measurement noise. Here we can ask, if the NN gives an advantage for the estimation of observables, by looking at the quotient $\text{RMS}_{\text{NN}}/\text{RMS}_{\text{Data}}$. Since a reduction of the RMS error of generic observables indicates that the state is faithfully represented, we use this ratio to quantify the approximation quality of our ansatz.

**Transverse field Ising model.** We start by benchmarking this method on ground states of a translation-invariant TFIM

$$H = \sum_{\langle i,j \rangle} J_{ij} \sigma_i^z \sigma_j^z - B \sum_i \sigma_i^x$$

with $J_{ij} = -J$ using the standard CNN. This serves as a proof of concept in an idealized scenario, as the translation invariance is directly encoded in the neural network. Since the comparison of a symmetrized network to an unsymmetrized reference is somewhat unfair, we refrain from enforcing any symmetries for the later examples.

Figure 2 shows the method being applied to small, i.e. MLE-suitable 1D states at the Ising-critical point. Our method achieves a reduction of infidelity by a factor 2-5, depending on system and dataset size. The figure shows one main trend: the network advantage shrinks for increased dataset size. This is also an expected result, as MLE has to outperform any variational approach in the limit of infinite dataset size. We generally see this behaviour for all studied systems.

In Fig. 3 we show the method being applied to a 4 × 4 lattice, which is not feasible for MLE anymore. The histograms show how enhancing the dataset using the NN can lead to a reduced variance of observable estimates, and thus a reduced error. For the network advantage, we see two trends: An increased advantage for small datasets, as well as an improved performance for smaller coupling strengths. The latter results in states closer to product states, which are easier for the network to learn. If $J/B$ becomes too large, training the network becomes unreliable on such small datasets and the advantage disappears.

**Noisy long-range interacting ion chain.** For a more experimentally motivated example, we look at ground states $|\psi_0\rangle$ of a 16-site, long-range interacting ion chain Hamiltonian, given by Eq. (7) with $J_{ij} = \frac{1}{(i-j)^2}$, open boundary conditions and small (3%) added dephasing noise. The target state is thus $\rho_{\text{Target}} = 0.97 |\psi_0\rangle \langle \psi_0| + 0.03 |\psi_{00}\rangle \langle \psi_{00}|$. As this is naturally a 1D system, we use the ARCNN. Here we study the NN advantage for correlation functions of increasing order. This is interesting, as higher order correlators are typically harder to estimate from samples, as the variance of the POVM-observable scales exponentially in correlation order. Thus, higher moments require a better approximation of the state, providing a sensitive benchmark for the quality of the state representation. Specifically, we look at powers $n$ of the Pauli-Z operator

$$C_n := \frac{1}{16 - n + 1} \sum_{i=1}^{16-n+1} \langle \sigma_i^x \sigma_{i+1}^z \ldots \sigma_{i+n-1}^z \rangle.$$  

(8)

Local observables, like the terms in Eq. (8) only depend on
the reduced density matrix of the subsystem they act on. Thus, these observables can in principle be estimated by performing MLE on this subsystem only. We show this “local MLE” applied to the bare dataset and to the NN-enhanced dataset in Fig. 4 in addition to the previously employed benchmarks. Using the NN-generated dataset, we see a reduction in RMS to a degree, that allows sampling for correlators of three orders higher, than what is possible with the bare dataset. When applying the local MLE to both bare and NN-generated dataset, this advantage is reduced significantly, but does not disappear. However, we emphasize that the CNN is on par with MLE, at a greatly reduced computational complexity. Notice also the peak in RMS at a correlation order of 2 for MLE, leading to an increased RMS compared to the bare dataset. We find this to be systematic, which is why we refrain from doing this comparison to local MLE in the remainder of this work, where second order correlation functions are of interest.

**Steady states of a driven dissipative 2D-system.** As a final system, we consider steady states of a $4 \times 4$ TFIM with spontaneous emission at $J = 1.25\gamma$ with open boundaries. $N_s = 1k$ (insets: 100k). Gray vertical line: Switch $\uparrow$ and $\downarrow$ in POVM for CNN. Upper: Observables according to Eq. (8) but summed only over system diagonal, Middle: residual RMS error, Lower: residual network RMS normalized to dataset RMS.

FIG. 4. Evaluating the ARCNN by evaluating observables from Eq. (8) and comparing to those obtained from plain dataset, on a length 16 ion chain with 3% dephasing noise, $J/B = 0.6$ and open boundaries, with a dataset size of $N_s = 10k$. Dashed lines show results of applying local MLE directly to dataset (orange) and to a neural network enhanced dataset (blue). Upper: Observables according to Eq. (8). Middle: residual RMS error, Lower: residual network RMS normalized to dataset RMS.

FIG. 5. Dissipative phase transition of a $4 \times 4$ TFIM with spontaneous emission at $J = 1.25\gamma$ with open boundaries. $N_s = 1k$ (insets: 100k). Gray vertical line: Switch $\uparrow$ and $\downarrow$ in POVM for CNN. Upper: Observables according to Eq. (11) but summed only over system diagonal, Middle: residual RMS error, Lower: residual network RMS normalized to dataset RMS.

to the standard CNN with dense output layer, since the system is two-dimensional. However, now no symmetries are enforced in the network. We use the Monte-Carlo wave function approach, to simulate the dynamics under the Lindblad master equation

$$\dot{\rho} = -i[H, \rho] + \gamma \sum_j \left( L_j \rho L_j - \frac{1}{2} \left\{ L_j^\dagger L_j, \rho \right\} \right)$$

with the Hamiltonian from Eq. (7) with antiferromagnetic coupling $J_{ij} = J$, $B < 0$ and

$$L_j = \sigma_j^- = \frac{1}{2} (\sigma_j^x - i\sigma_j^y),$$

until the steady state is reached. We consider the density matrix that is obtained using 1000 pure-state trajectories as our exact target. This system undergoes a dissipative phase transition [40] which is visible as a peak in the correlation length

$$\xi_z^2 = \sum_{i,j} |\mathbf{r}_i - \mathbf{r}_j|^2 \left( \langle \sigma_i^z \sigma_j^z \rangle - \langle \sigma_i^z \rangle \langle \sigma_j^z \rangle \right).$$

We show this phase transition as obtained by computing the correlation length on the 1D diagonal of the 2D lattice, once directly from the training data, as well as
from a NN-enhanced dataset in Fig. 5. The network is able to capture the phase transition, as a peak in the correlation length at $|B/\gamma| \approx 2$ is clearly visible. At the dashed gray line, we exchange $\uparrow$ and $\downarrow$ in the POVM that the CNN uses, ensuring that the target state does not contain exact zeros in its POVM distribution. For small $|B|$, the steady state tends towards an eigenstate of the observable in question. Thus the variance of sampling this observable is significantly reduced and the network has no advantage here. For the limit of large $|B|$ we see a huge variance in the sampled correlation length, which the CNN trades for a small bias. The overall effect is that the CNN bias and CNN variance lead to a significantly smaller RMS error as compared to the bare dataset. Notice that this bias also shrinks with increasing dataset size (Fig. 5 (insets)).

Depending on the observable of interest, the bias can have a more severe effect than depicted. When computing the correlation length over the 1D diagonal, as in Fig. 5, the corresponding sum in Eq. 11 is a weighted average of $\langle \sigma_i \sigma_j \rangle = 6$ connected correlators of the form $\langle \sigma_i \sigma_j \rangle - \langle \sigma_i \rangle \langle \sigma_j \rangle$. After sampling, one may consider each of these connected correlators as a random variable with a variance and a bias. For the plain dataset, this bias is of course zero. However, when evaluating Eq. 11 over the entire lattice, the sum contains $\binom{16}{2} = 120$ terms, with roughly similar variance and bias. For the sampled case, by simple addition of probability distributions, the variance of the latter observable is thus reduced by a factor of $\sqrt{120/6} \approx 4.5$ compared to the former one. Due to the bias, the network is not able to make use of this self averaging effect, resulting in a significantly reduced advantage. We show this full system correlation length in the appendix. Thus the network advantage is the largest, if one is interested in the expectation values of individual correlators, and might be smaller if large sums over many similarly distributed correlators are involved as the bias inherent to the variational approach becomes more statistically significant for the averaged results.

**Discussion.** Motivated by the proven superior representation capabilities of CNNs [31], we explored the application of two different CNN architectures to NN-QST tasks, i.e. the reconstruction of pure and mixed quantum states. We especially found the autoregressive CNN to be extremely versatile, due to its great expressivity, exact sampling and stability during training. For a broad range of experimentally relevant scenarios, including pure and dephased ground states, as well as steady states, we presented quantitative comparisons to traditional schemes. This showed a significant advantage over MLE or, for system sizes inaccessible to MLE, direct sampling of local observables from an IC-POVM dataset. Although we demonstrated the described properties on synthetic data, the method may be readily applied to real quantum simulation experiments.

Like any variational approach, which benefits from restricting the state space to a representable subspace, this scheme is subject to potential bias. The effect of such bias intimately depends on the estimated observable, the approximated state, as well as the sample size, as discussed above. This makes it necessary to a priori validate the method to ensure generalization of a given model to the situation at hand.

We emphasize that the proposed method can harness the strengths of any variational function approximator, thus directly profiting from the rapid development of ever more expressive architectures in the machine learning community. New network architectures can only enhance the state space that is covered by NN-QST, motivating further research in this area. For future projects it would be interesting to compare many of the NN-QST schemes, to methods like shadow tomography [5, 6], which explicitly claim superiority over NN-QST.

**Acknowledgments.** We thank J. Carrasquilla and M. Schmitt for discussions. Monte-Carlo wave function trajectories were computed using the QUTIP library [11]. The JAX library [12] as well as the FLAX framework [13] were used to build the neural network models and training algorithms. This work is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy EXC2181/1-390900948 (the Heidelberg STRUCTURES Excellence Cluster) and within the Collaborative Research Center SFB1225 (ISOQUANT). This work was partially financed by the Baden-Württemberg Stiftung gGmbH. The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant no INST 40/575-1 FUGG (JUSTUS 2 cluster). The authors gratefully acknowledge the Gauss Centre for Supercomputing e.V. (www.gauss-centre.eu) for funding this project by providing computing time through the John von Neu mann Institute for Computing (NIC) on the GCS Supercomputer JUWELS [44] at Jülich Supercomputing Centre (JSC).

**Appendix**

**Details on network architectures.** We employ two distinctly different network architectures. The first is the "standard" CNN in Fig. 1 (top left), which is fed a one-hot encoded vector of single-site outcomes and performs $L$ convolutions with kernels of size $K$. Boundary conditions are either open or periodic, depending on the symmetries of the target state. A final layer turns the network output into a single scalar. Two common options are a "dense" layer, or a "product" layer, the latter resulting in network outputs of the form $P_{NN}(a) = e^{\sum_{\text{last layer}}}$. This results in an $nn$-normalized probability distribution. Therefore, before each training step, a Monte Carlo estimate of the normalization constant has to be performed. For normalization, we
generate as many uniform POVM samples as there are samples within each training batch. For translation invariant states, we use the "product" output layer and the "dense" output layer otherwise.

The second architecture is a 1D autoregressive CNN (ARCNN) \cite{6, 37} in Fig. 1 (top right). This architecture makes use of the autoregressive property, which states that a probability of multiple variables can be partitioned into a product of conditionals: \( P(a) = P(a_1)P(a_2|a_1)\ldots P(a_N|a_1,a_{N-1}) \). The autoregressive CNN only differs from the standard CNN in the boundary conditions and in the fact that the physical dimension is shifted by one site, so that the "last" site is not used as an input. The outputs are fed into a softmax layer, which results in four conditional probabilities to be returned from each site. The exactly normalized probabilities may be computed by passing a POVM outcome through the network and post-selecting the correct conditional probability at each site. Exact samples can be drawn from this distribution, by passing zeros through the network, sampling the first site, passing this outcome through the network and sampling the next site, etc. This results in \( N \) forward passes in order to generate one exact POVM sample, at the benefit of circumventing Markov chains.

In addition to exact normalization and sampling, we empirically find this network architecture to be substantially easier to train, and the training algorithm to converge for a wider variety of states. There is no simple generalization of this approach to higher dimensional systems, however approaches like \cite{36} have been proposed.

The hyperparameter configurations used for producing the data shown in each figure are summarized in Tab. 1.

**Self-averaging can reduce network advantage.** As explained in the main text, the effect of a bias, that the network might introduce for a given observable may be amplified when one is interested in observables which contain sums over many independent observables. In this scenario the network-bias inhibits any positive influence of possible self averaging effects. We demonstrate this, by computing the correlation length (11) by summing over all pairs of lattice sites, as opposed to only summing over a diagonal as in Fig. 5. The result is shown in Fig. 6. The sampled observables self-average, resulting in a smaller statistical error at fixed sample size, while the network systematically overestimates the correlation length for fields greater than the critical field.

---

1. A. I. Lvovsky, Iterative maximum-likelihood reconstruction in quantum homodyne tomography, *J. Opt. B: Quantum Semiclass. Opt.* 6, S556
2. R. Blume-Kohout, Optimal, reliable estimation of quantum states, *New J. Phys.* 12, 043034 (2010)
3. C. Harney, S. Pirandola, A. Ferraro, and M. Paternostro, Entanglement classification via neural network quantum states, *New J. Phys.* 22, 045001 (2020)
4. C. Harney, M. Paternostro, and S. Pirandola, Mixed state entanglement classification using artificial neural networks, *New J. Phys.* 23, 063033 (2021)
5. H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, *Nat. Phys.* 16, 1050 (2020)
6. G. Struchalin, Y. A. Zagorovskii, E. Kovlakov, S. Straupe, and S. Kulik, Experimental estimation of quantum state properties from classical shadows, *PRX Quantum* 2 (2021)
7. T. Baumgratz, D. Gross, M. Cramer, and M. B. Plenio, Scalable reconstruction of density matrices, *Phys. Rev. Lett.* 111, 020401 (2013)
8. M. Cramer, M. B. Plenio, S. T. Flammia, R. Somma, D. Gross, S. D. Bartlett, O. Landon-Cardinal, D. Poulin, and Y.-K. Liu, Efficient quantum state tomography, *Nat. Commun.* 1, 149 (2010)
9. B. P. Lanyon, C. Maier, M. Holzäpfel, T. Baumgratz, C. Hempel, P. Jurcevic, I. Dhand, A. S. Buyskikh, A. J. Daley, M. Cramer, M. B. Plenio, R. Blatt, and C. F. Roos, Efficient tomography of a quantum many-body system, *Nat. Phys.* 13, 1158 (2017)
10. D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum state tomography via compressed sensing,
Table I. Network architectures used for each figure. In all cases we used a tanh-activation function, $N$ features per layer for $N$ particles, and an exponential output function. $[x]$ denotes the nearest integer function and $\lceil x \rceil$ the ceiling function.

| Figure | Network architecture | Network boundary conditions | Num. Layers $L$ | Kernel size $K$ | Num. Parameters |
|--------|----------------------|-----------------------------|----------------|----------------|-----------------|
| 2      | Small 1D TFIM        | CNN                         | Circular       | $\lceil \sqrt{N} \rceil$ | $\lceil \sqrt{N} \rceil + 1$ | 73 - 673 |
| 3      | Larger 2D TFIM       | CNN                         | Circular       | 2              | 2               | 1329            |
| 4      | Ion Chain            | ARCNN                       | Open           | 3              | 6               | 3572            |
| 5      | Steady States        | CNN                         | Open           | 2              | 3               | 3169            |

Phys. Rev. Lett. 105, 150401 (2010)

[11] C. A. Riofrío, D. Gross, S. T. Flammia, T. Monz, D. Nigg, R. Blatt, and J. Eisert, Experimental quantum compressed sensing for a seven-qubit system, Nat. Commun. 8, 15305 (2017)

[12] G. Tóth, W. Wieczorek, D. Gross, R. Krischek, C. Schwemmer, and H. Weinfurter, Permutationally invariant quantum tomography, Phys. Rev. Lett. 105, 250403 (2010)

[13] T. Moroder, P. Hyllus, G. Tóth, C. Schwemmer, A. Nigebaum, S. Gaile, O. Gühne, and H. Weinfurter, Permutationally invariant state reconstruction, New J. Phys. 14, 105001 (2012)

[14] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science 355, 602 (2017)

[15] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Neural-network state tomography, Nat. Phys. 14, 447 (2018)

[16] G. Torlai and R. G. Melko, Machine-learning quantum states in the NISQ era, Annu. Rev. Condens. Matter Phys. 11, 325 (2020)

[17] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, Neural Netw 2, 359 (1989)

[18] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Netw 4, 251 (1991)

[19] O. Sharir, A. Shashua, and G. Carleo, Neural tensor contractions and the expressive power of deep neural quantum states, arXiv:2103.10293 (2021)

[20] D.-L. Deng, X. Li, and S. Das Sarma, Quantum entanglement in neural network states, Phys. Rev. X 7, 021021 (2017)

[21] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, Reconstructing quantum states with generative models, Nat. Mach. Intell. 1, 155 (2019)

[22] G. Torlai, G. Mazzola, G. Carleo, and A. Mezzacapo, Precise measurement of quantum observables with neural-network estimators, Phys. Rev. Research 2, 022060 (2020)

[23] F. Chi, F. Ginsparg, F. Wu, J. Carrasquilla, P. L. McMahon, and E.-A. Kim, Attention-based quantum tomography, arXiv:2106.12469 (2020)

[24] A. W. R. Smith, J. Gray, and M. S. Kim, Efficient quantum state sample tomography with basis-dependent neural-networks, PRX Quantum 2, 020348 (2021)

[25] A. Melkani, C. Geeting, and F. Nori, Eigenstate extraction with neural-network tomography, Phys. Rev. A 102, 022412 (2020)

[26] H. Huang and H. Situ, Investigating reconstruction of quantum state distributions with neural networks, Eur. Phys. J. Plus 136, 204 (2021)

[27] A. M. Palmieri, E. Kovlakov, F. Bianchi, D. Yudin, S. Straupe, J. D. Biamonte, and S. Kulik, Experimental quantum neural network enhanced quantum tomography, NP Quantum Inf. 6, 20 (2020)

[28] G. Torlai, B. Timar, E. P. L. van Nieuwenburg, H. Levine, A. Ouran, A. Keesling, H. Bernien, M. Greiner, V. Vuletić, M. D. Lukin, R. G. Melko, and M. Endres, Integrating neural networks with a quantum simulator for state reconstruction, Phys. Rev. Lett. 123, 230504 (2019)

[29] M. Neugebauer, L. Fischer, A. Jäger, S. Cziesche, S. Jochim, M. Weidemüller, and M. Gärttner, Neural network quantum state tomography in a two-qubit experiment, Phys. Rev. A 102, 042604 (2020)

[30] E. S. Tiunov, V. V. Tiunov (Vyorova), A. E. Ulanov, A. I. Lvovsky, and A. K. Fedorov, Experimental quantum homodyne tomography via machine learning, Optica 7, 448 (2020)

[31] Y. Levine, O. Sharrir, N. Cohen, and A. Shashua, Quantum entanglement in deep learning architectures, Phys. Rev. Lett. 122, 065301 (2019)

[32] J. Carrasquilla, D. Luo, F. Pérez, A. Milsted, B. K. Clark, M. Volkovs, and L. Aolita, Probabilistic simulation of quantum circuits using a deep-learning architecture, Phys. Rev. A 104, 032610 (2021)

[33] M. Reh, M. Schmitt, and M. Gärttner, Time-dependent variational principle for open quantum systems with artificial neural networks, arXiv:2104.00013 (2021)

[34] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980 (2017)

[35] C. A. Fuchs and R. Schack, Quantum-bayesian coherence, RMP 85, 1693–1715 (2013)

[36] O. Sharir, Y. Levine, N. Wies, G. Carleo, and A. Shashua, Deep autoregressive models for the efficient variational simulation of many-body quantum systems, Phys. Rev. Lett. 124, 020503 (2020)

[37] S.-H. Lin and F. Pollmann, Scaling of neural-network quantum states for time evolution, arXiv:2104.10696 (2021)

[38] B. P. Lanyon, P. Jurcevic, M. Zwerger, C. Hempel, E. A. Martinez, W. Dür, H. J. Briegel, R. Blatt, and C. F. Roos, Measurement-based quantum computation with trapped ions, Phys. Rev. Lett. 111, 210501 (2013)

[39] W. L. Tan, P. Becker, F. Liu, G. Pagano, K. S. Collins, A. De, L. Feng, H. B. Kaplan, A. Kyriienko, R. Lundgren, W. Morong, S. Whitesitt, A. V. Gorshkov, and C. Monroe, Observation of domain wall confinement and dynamics in a quantum simulator, Nat. Phys. 17, 8
[40] J. Jin, A. Biella, O. Viyuela, C. Ciuti, R. Fazio, and D. Rossini, Phase diagram of the dissipative quantum Ising model on a square lattice, Phys. Rev. B 98, 241108.

[41] J. Johansson, P. Nation, and F. Nori, QuTiP 2: A python framework for the dynamics of open quantum systems, Comput. Phys. Commun. 184, 1234 (2013).

[42] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, JAX: composable transformations of Python+NumPy programs (2018).

[43] J. Heek, A. Levskaya, A. Oliver, M. Ritter, B. Rondepierre, A. Steiner, and M. van Zee, Flax: A neural network library and ecosystem for JAX (2020).

[44] Jülich Supercomputing Centre, JUWELS: Modular Tier-0/1 Supercomputer at the Jülich Supercomputing Centre, JLSRF 5 (2019).