Effect of Loss Function on Supervised Learning of Quantum Many-Body States

Oisín Morrison, Nada Alghamdi and Mohamad Fakhouri
École Polytechnique Fédérale de Lausanne, Switzerland
Project hosted by Prof. Giuseppe Carleo, CQSL, Institute of Physics, EPFL

***Abstract*—Simulating the dynamics of quantum many-body systems using neural networks is a difficult task due to the exponential scaling in the number of parameters of the network with time required to obtain reasonable approximations. Recent work has shown that this scaling occurs independently of network architecture; however, the effects of loss functions on the performance of the network are unknown. In this work, we study the effect of different loss functions on the performance of a certain network in modelling the dynamics of a quantum many-body system at different times. Our work shows that loss functions can have a major impact on the ability of our network to model the dynamics of quantum many-body states.**

I. Introduction and Background

The use of artificial neural networks (ANNs) has emerged as a new promising approach to simulate the dynamics of quantum many-body systems [1][2]. This approach is based on the universal approximation theorem which states that any continuous function can be approximated by some neural network architecture [3]. However, the simulation of quantum many-body systems is difficult as quantum systems become increasingly entangled with time. In fact, it has been found that simulating the dynamics of multi-particle quantum systems requires an exponential scaling in the number of parameters of the network with time regardless of model architecture [2], making it computationally expensive. However, there is currently a lack of research on the role of the loss function on the accuracy of the network. This is vital to test since learning is minimizing the objective function, or the loss function. In this work, we examine this issue by comparing the performance of 7 different loss functions on a convolutional neural network (CNN) architecture.

II. Methods

A. Data Generation

To create the data for the supervised learning problem, state vectors are generated with NetKet [4] using the same set-up as Lin *et al.* [2]. Specifically, an exact time evolution of a chain of $N = 20$ spins is considered, and state vectors are generated for times up to $t = 5$. An x-polarised initial product state $|\Phi_0\rangle = \Pi_i |\rightarrow\rangle_i$ is evolved using the time dependent Schrodinger equation with the Hamiltonian of the 1D quantum Ising model, with transverse (strength g), longitudinal (strength h) fields and an interaction term (strength k).

$$\hat{H} = -J\left[\sum_{j=1}^{N-1}\left(\hat{\sigma}_j^z\hat{\sigma}_{j+1}^z + k\hat{\sigma}_j^x\hat{\sigma}_{j+1}^x\right) + \sum_{j=1}^{N}\left(g\hat{\sigma}_j^x + h\hat{\sigma}_j^z\right)\right]. \tag{1}$$

We consider the case of a strong quench near the critical point, where in the Hamiltonian eq. 1, $g = 1$, $h = 0$ and $k = 0.25$. We apply periodic boundary conditions since we are interested in approximating the system in the thermodynamic limit. This also preserves the translational invariance of the problem i.e. at any time $\Phi(\sigma_1, ..., \sigma_{n-1}, \sigma_n) = \Phi(\sigma_n, \sigma_1 ..., \sigma_{n-1})$.

B. Model Architecture

In this work, we employ a simple CNN architecture. Previous work has found that for quantum many body systems, a fully connected network with one hidden layer performs better than an ANN with multiple layers and additionally a CNN network with multiple hidden layers performs better than a fully connected network or a single layer CNN [5]. It is worth noting that network architectures do not play a major role in the exponential scaling of parameters with time [2], so the network architecture used in this work is not of the utmost importance considering the problem we are addressing.

Our network is composed of three convolution blocks, each of which contains a convolutional layer with rectified linear unit (ReLU) activation followed by an average pooling layer. After the the convolutional blocks, we flatten and then use three fully connected layers, leading up to a final layer that outputs two real values that correspond to the real and complex value of the wave function. For learning, the logarithm of the wave functions are considered since this makes it easier for the model to learn amplitudes as their values can vary by several orders of magnitude [6]. The network is entirely real-valued, so the wave functions were decomposed into two values, the real and imaginary parts, given as $\mathrm{Re}\{\ln\Phi\}$ and $\mathrm{Im}\{\ln\Phi\}$, respectively. The network then outputs the estimates $\mathrm{Re}\{\ln\Psi^{NN}\}$ and $\mathrm{Im}\{\ln\Psi^{NN}\}$. The weights of the layers were initialized using a Glorot uniform initialiser, and biases were initialised with zeros. The entire network is shown in Figure 1.

We furthermore take advantage of some properties of the physics in the design of our network. Since periodic

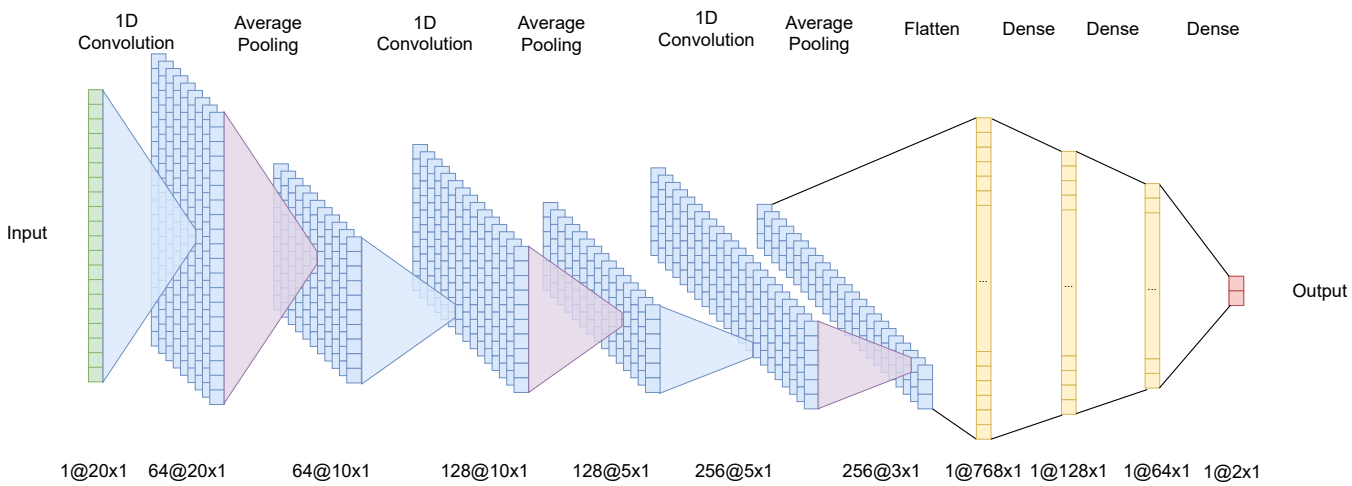

Figure 1: The CNN architecture employed in this work. Convolutions are done with ReLU activation and circular padding.

boundary conditions are at play, circular padding is applied to the convolutional layers. This ensures that the network is also translationally invariant, and furthermore avoids strange behaviour occurring at the boundary [7]. We also enforce a kernel size of 20 for first convolutional (equal to the number of basis states). This means that the first convolution considers all possible orderings of the basis states.

Average pooling was performed with strides of 2 and pool sizes of 3. All other parameters are depicted in Figure 1. The total number of learnable parameters is 305,154.

C. Hyperparameters

For our choice of an optimizer, we selected the Adam optimiser with an initial learning rate of 10^{-3} (which is in line with Lin *et al.* [2]). Work by Saito *et al.* has also found that for the quantum many body ground state problem, the Adam optimizer allows faster convergence than a simple steepest-descent method [5]. Early stopping is employed as a regularization technique, in line with the work by Westerhout *et al.* [8]. A batch size of 4096 was used since 1) some of the loss functions we consider such as the negative log overlap require large batch sizes [2] and 2) the training units contained enough memory to store the large batches.

D. Training Procedure

This work focuses specifically on examining the learnability of the quantum system; hence, some general machine learning protocols do not apply. For example, there is no need to ensure that our training and testing datasets are independent since we are solely interested in the expressivity of the given neural network for various loss functions (i.e. how well it can learn the quantum states). In addition, since our data is not observed but is in fact calculated, we sample our training data according to the probability distribution given by the amplitudes of the wavefunctions i.e. $\boldsymbol{\sigma_i} \sim |\Psi(\boldsymbol{\sigma_i})|^2$. This sampling means that our losses are now expectation values over all states. To simplify notation, we adopt the shorthand $\Psi_i = \Psi(\boldsymbol{\sigma_i})$.

A training size of 524,288 (half the size of the full dataset) was used, and a validation set of size 349,525 (a third of the size of the full dataset) was sampled uniformly from the data. The training loop was then run using these two datasets. For each time step we train our network with exactly the same hyperparameters. We consider the following times: $t \in \{0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5\}$. Thus, for every loss function (see Sec. III) we train 11 models (one for each of the 11 different time steps). To ensure models learned sufficiently from training, a minimum of 75 epochs were conducted before possible early stopping.

E. Metrics

We consider three metrics that are of special importance to the physical problem.

1) The infidelity, $1 - \mathcal{F}$, which is the difference between the network prediction and the true wave function. The fidelity, $\mathcal{F}$, expresses the similarity between the prediction and the true wavefunction and is given by

$$\mathcal{F} = \frac{|\left\langle \Phi \middle| \Psi^{NN} \right\rangle|^2}{\langle \Phi | \Phi \rangle \left\langle \Psi^{NN} \middle| \Psi^{NN} \right\rangle} \tag{2}$$

2) The von Neumann entropy, S_{vN}, which measures the degree of entanglement for quantum states.
3) The expectation value of the spin is given by

$$\langle S_x \rangle = \frac{\hbar}{2} \langle \sigma_x \rangle = \frac{\hbar}{2} \langle \Psi | \sigma_x | \Psi \rangle \,, \tag{3}$$

where Ψ is the normalized wavefunction. Specifically, we look at this expectation at the middle of the chain $\left\langle \sigma^x_{L/2} \right\rangle$.

It is to be expected that the saturation of the the von Neumann entropy and the deviation of the expectation value follow a similar trend as the exponential growth in the fidelity [2].

III. Loss Functions

A number of varying loss functions are considered in the literature, and we consider some additional variations of our own in this work as well. A total of 7 loss functions are examined.

A. MSE of the Wavefunctions (`mse`)

The mean squared error (MSE) loss function is ubiquitous in machine learning tasks, and is applicable in this case also. It has been implemented previously by Kochkov *et al.* [9].

$$\mathcal{L}_1 = \sum_i |\Phi_i - \Psi_i^{NN}|^2 \tag{4}$$

B. MSE of the Logarithm of the Wavefunctions (`mse_log`)

Since we work with the logarithm of the wavefunctions in the network, we consider taking the MSE of the logarithm of the wavefunctions.

$$\mathcal{L}_2 = \sum_i |\ln \Phi_i - \ln \Psi_i^{NN}|^2 \tag{5}$$

C. L2 Distance of the Logarithm of Probability Amplitudes (`l2_log_ampl`)

Another loss function which has been used by Westerhout *et al.* [8] is the MSE of the logarithm of the probability amplitudes of the wavefunctions.

$$\mathcal{L}_3 = \sum_i \left(\ln |\Phi_i| - \ln |\Psi_i^{NN}|\right)^2 \tag{6}$$

This loss function is similar to `mse_log` - the difference is that in this loss function we minimise only the amplitude of the wavefunction and thus the phases are not minimised.

D. MAE of the Wavefunctions (`mae`)

Another commonly used loss function in machine learning is mean absolute error (MAE), which we can employ analogously to MSE.

$$\mathcal{L}_4 = \sum_i \begin{array}{l}\big[|\mathrm{Re}\{\Phi_i\} - \mathrm{Re}\{\Psi_i^{NN}\}| \\ +|\mathrm{Im}\{\Phi_i\} - \mathrm{Im}\{\Psi_i^{NN}\}|\big]\end{array} \tag{7}$$

E. MAE of the Logarithm of the Wavefunctions (`mae_log`)

$$\mathcal{L}_5 = \sum_i \begin{array}{l}\big[|\mathrm{Re}\{\ln \Phi_i\} - \mathrm{Re}\{\ln \Phi_i^{NN}\}| \\ +|\mathrm{Im}\{\ln \Psi_i\} - \mathrm{Im}\{\ln \Psi_i^{NN}\}|\big]\end{array} \tag{8}$$

F. Negative Log Overlap (`neg_log_overlap`)

A very common loss function is the negative log overlap of the training and test states. This is often used and is available in NetKet [4].

$$\mathcal{L}_6 = -\ln \frac{\left(\sum_i \Psi_i^{*NN}\Phi_i\right)\left(\sum_i \Phi_i^*\Psi_i^{NN}\right)}{\left(\sum_i \Psi_i^{*NN}\Psi_i^{NN}\right)\left(\sum_i \Phi_i^*\Phi_i\right)} \tag{9}$$

We also note that computing this loss on the entire dataset would yield the fidelity $\mathcal{F}$.

G. Joint Loss Function (`joint`)

A new loss function was proposed by Lin *et al.* [2] based off of the Kullback–Leibler (KL) divergence. To a first order approximation, the function is equivalent to the negative overlap. However, the function is unbounded and therefore requires normalised wavefunctions. We do not consider normalisation in our case, thus instead an absolute value is taken, giving the following approximation:

$$\mathcal{L}_7 = \mathcal{L}_{\widetilde{KL}} + \mathcal{L}_\theta, \tag{10}$$

where

$$\mathcal{L}_{\widetilde{KL}} \approx \sum_i |2\mathrm{Re}\{\ln \Phi_i - \ln \Psi_i^{NN}\}| \tag{11}$$

and

$$\mathcal{L}_\theta \approx \sum_i \mathrm{dist}(\mathrm{Im}\{\ln \Phi_i\}, \mathrm{Im}\{\ln \Psi_i^{NN}\}) \tag{12}$$

with

$$\mathrm{dist}(\theta_1, \theta_2) = (\cos\theta_1 - \cos\theta_2)^2 + (\sin\theta_1 - \sin\theta_2)^2 \tag{13}$$

eee

IV. Results and Discussion

Histories for the models were monitored after training to verify losses were decreasing and converging correctly. By observing the losses while training, we see that in general learning the quantum states becomes harder (losses are higher) as the quantum system evolves through time (Figure 2).

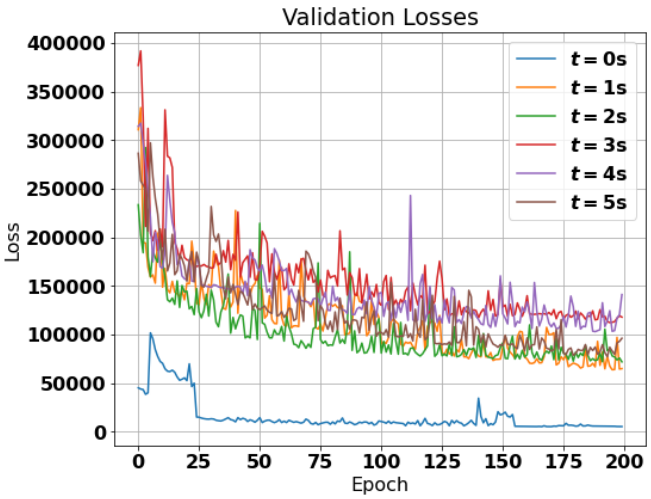

Figure 2: The validation loss across epochs when training using the `joint` loss for $t \in \{0, 1, 2, 3, 4, 5\}$.

After training, we compute the three metrics: the infidelity $1 - \mathcal{F}$, the expectation of the spin $\left\langle \sigma^x_{L/2} \right\rangle$ and the von Neumann entropy S_{vN} on the entire dataset. It is vital to compute these metrics on the full data set because from the physics point of view, they do not make sense unless they are computed on all the wavefunctions in the Hilbert space. Results for all of these metrics are shown in Figure 3.

Comparing the infidelity obtained by the different losses, in general the lowest fidelity corresponds to

neg_log_overlap. However, this loss has also high variation in the infidelity.

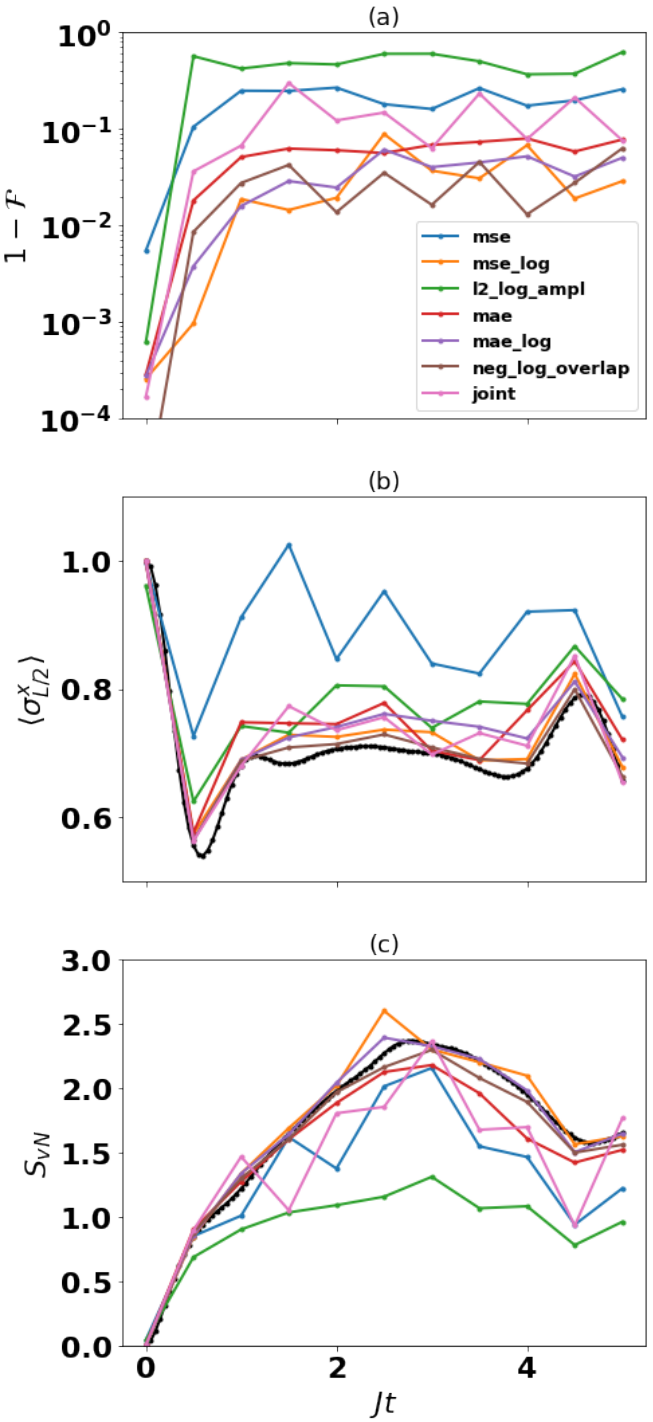

Figure 3: We show the results of varying loss functions on three metrics - namely (a) the fidelities, (b) the expectation value of the spin and (c) the von Neumann entropy. Results were evaluated at 11 times: $t \in \{0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5\}$. The exact solution is shown in black.

The other two metrics are nonetheless approximated quite well with it. This agrees with our intuition as this loss function coincides with the fidelity if one evaluates it over all data. It is interesting to also note that both mse_log and mae_log performed better than mse and mae. This suggests that working with the log of the wavefunctions yields better results than working with the wavefunctions themselves. However, mse and mae are more stable as they don't fluctuate as much as mse_log and mae_log. On the other hand, l2_log_ampl performed worse than all the other variations of MSE and MAE that we have, which may be indication of the importance of learning the phase as well as the amplitude of the wavefunction. The growth in entropy and expectation value shows similar trend with respect to the different loss functions as the fidelity where again the best performance comes from neg_log_overlap. The only exception for this pattern is l2_log_ampl performs actually better than mse with respect to the expectation value. This is interesting to note since it shows that some losses are better able to follow certain dynamics better than others. For example, it is very clear that mse does a poorer job than l2_log_ampl at following the expectation value of the spin, but it performs better than l2_log_ampl in following the von Neumann entropy.

Lin *et al.* [2] studied the same system with different network architectures. It is interesting to compare the results putting in mind that the our architecture does not match theirs. We note that our joint loss function is a slight modification of their R_joint. We take the absolute value of the real part to bound the loss while they rely on normalizing the wave function in their network to bound it. It is interesting to note that this loss function performed relatively well for us, with values of infidelity ranging between $10^{-1.5}$ and 10^{-2}. We do observe fluctuations in the infidelity values, but not in the expectation value or the von Neumann entropy. To conclude, we find that by fixing our network structure and experimenting with different losses we find a difference of two orders of magnitude in fidelity between our best and worst performance for the same network.

V. Conclusion

In this work, we study the effect of various loss functions on the performance of a given network in modelling the dynamics of a quantum many-body system at different times. We find that choosing an appropriate loss function is vitally important in determining how well a neural network can learn to approximate complicated states. For our network, we find that the negative log overlap loss function performs the best overall; we also find that using the logarithm of the true wavefunction in the loss gives better performance. It would be interesting in future work to test the effect of the batch size and the sampling method on the expressivity of the neural network, and to examine the interplay between model architecture and loss functions.

REFERENCES

[1] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," *Science*, vol. 355, no. 6325, pp. 602–606, 2017.

[2] S.-H. Lin and F. Pollmann, "Scaling of neural-network quantum states for time evolution," *arXiv preprint arXiv:2104.10696*, 2021.

[3] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural networks*, vol. 6, no. 6, pp. 861–867, 1993.

[4] G. Carleo, K. Choo, D. Hofmann, J. E. Smith, T. Westerhout, F. Alet, E. J. Davis, S. Efthymiou, I. Glasser, S.-H. Lin *et al.*, "Netket: A machine learning toolkit for many-body quantum systems," *SoftwareX*, vol. 10, p. 100311, 2019.

[5] H. Saito and M. Kato, "Machine learning technique to find quantum many-body ground states of bosons on a lattice," *Journal of the Physical Society of Japan*, vol. 87, no. 1, p. 014001, 2018.

[6] F. Vicentini, D. Hofmann, A. Szabó, D. Wu, C. Roth, C. Giuliani, G. Pescia, J. Nys, V. Vargas-Calderon, N. Astrakhantsev, and G. Carleo, "Netket 3: Machine learning toolbox for many-body quantum systems," 2021.

[7] A. Alguacil, W. G. Pinto, M. Bauerheim, M. C. Jacob, and S. Moreau, "Effects of boundary conditions in fully convolutional networks for learning spatio-temporal dynamics," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 102–117.

[8] T. Westerhout, N. Astrakhantsev, K. S. Tikhonov, M. I. Katsnelson, and A. A. Bagrov, "Generalization properties of neural network approximations to frustrated magnet ground states," *Nature communications*, vol. 11, no. 1, pp. 1–8, 2020.

[9] D. Kochkov and B. K. Clark, "Variational optimization in the ai era: Computational graph states and supervised wave-function optimization," *arXiv preprint arXiv:1811.12423*, 2018.