Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Physics

Machine-Learning Time-Dependant Density Functional Theory on a Lattice

Aaron Commons

Supervised by Professor Stefano Sanvito

April 10, 2023

Submitted in partial fulfilment of the requirements for the degree of

B.A. (Mod) in Theoretical Physics

Declaration

I hereby declare that this research project is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed: Aron Connor Date: 10/04/2023

Abstract

Machine learning has emerged as an important tool in the treatment of the quantum many-body problem. Along with density functional theory, it serves as an approach studying the behaviour and properties of systems of interacting particles. In this project, we aim to construct a neural network to describe the time-propagation of a quantum many-body system on a lattice. Electronic interactions on a lattice are simulated using the spinless Hubbard model. Exact diagonalisation allows datasets of the exact time evolution of a small many-body system to be generated, upon which a network is trained to make predictions on the future evolution of new initial states. Lattice density functional theory offers the theoretical foundation for studying the system in the compressed representation of average site occupation to avoid the exponential scaling of the wavefunction, allowing larger systems to be studied. We demonstrate for a system fully represented by its density matrix, the network is capable of making very accurate predictions of its future state given its current state only. Its evolution over a time interval is reproduced with high precision using autoregression. Expressing the system by its vector of site occupation, the network required a collection of its previous timesteps to make accurate predictions for its future evolution. We showed for a region of intermediate interaction strength, the model made the least accurate predictions compared to the high and low interaction strength limits. The research iterates the capabilities of machine learning approaches in understanding the dynamics of many-body systems.

Acknowledgements

I would like to take this opportunity to thank all the people who have helped me in the completion of this research project, without whom it couldn't have been possible.

I would firstly like to thank my supervisor Prof. Stefano Sanvito for his continuous guidance and invaluable suggestions throughout the process. I am extremely grateful for the time he has taken to provide feedback on presented work and his willingness to answer any question I had along the way.

I would also like to express my thanks to Dr. Rajarshi Tiwari for dedicating the time to sit down with me and work through any technical or coding issues that I encountered throughout the process. Learning from his knowledge and expertise played a crucial role in the completion of this project.

Finally, I want to express my gratitude to my friends and family, especially my parents, whose constant support and encouragement have helped me enormously throughout my undergraduate degree.

Contents

1. Introduction

One of the most important problems in condensed matter physics is that of the quantum many-body system, essential for understanding the emerging properties of systems of interacting particles. However, any exact computations involving the many-body wavefunction quickly becomes a numerically impossible task due to the exponential scaling of the Hilbert space needed to describe a many-body system. Therefore, one of the primary goals of research in this field is to explore methods of dimensional reduction of the many-body wavefunction, such as the density-matrix renormalization group [1], enabling properties of larger systems to be analysed.

Density functional theory (DFT) is one method of addressing the exponentially scaling limitations of many-body systems, based on the theorems of Hohenberg and Kohn [2]. It establishes a one-to-one correspondence between the electron density and the external potential defining the Hamiltonian of the system, hence determining its solutions. It has been shown that an accurate approximate solution for a many-body system can be obtained using an appropriate electron density [3], a quantity which scales linearly with system size for lattice models.

In this project, we look at the time evolution of a many-body system on a lattice simulated by the spinless Hubbard model [4]. In particular, we investigate the extension of DFT to the time domain, which states that the time evolution of a many-body system can be studied through the time evolution of its average site occupation vector [5]. One does not know the precise form of this time propagator and so, it needs to be approximated. Having seen the success of machine learning and neural networks in real-world industry problems [6, 7] and tasks such as image recognition [8], these approaches have been adopted by condensed matter physicists in the treatment of the many-body problem. It has shown promising applications in numerous areas of quantum many-body systems, such as dimensional reduction [9], time propagation [10], and learning the Hamiltonian of quantum interacting systems [11]. Neural networks have also been applied to constructing exact functionals for the Hubbard model [12] and has shown promising results for networks trained on small systems applied to larger system evaluations [13]. In this report, we will take a machine learning approach to approximate the time propagator of a many-body system described by the spinless Hubbard Hamiltonian, by constructing a neural network which learns its evolution.

The report will start by introducing the necessary background theory. Firstly, the Hubbard Hamiltonian for modelling interacting systems of electrons on a lattice is derived from the second quantization of the classical Hamiltonian. We describe exact diagonalisation (ED), a technique for finding numerically exact solutions of the Hubbard model, and apply it to a small many-body system. We will then provide the theoretical foundations of DFT for an arbitrary system of interacting electrons and extend the formalism to lattice models in the time domain. The report will then briefly outline the basics of machine learning and neural networks, their application in learning the time evolution of a quantum system and how to generate a uniform sample of initial states suitable for machine learning.

We finally present the findings of the research. For a system described by the full representation of the density matrix, the learning of the time evolution only requires the knowledge of the current state to make accurate predictions about future states. For a reduced representation, as in density functional theory, the evolution is no longer local in time, and we require a history of previous time steps for the system to make precise predictions into the future. Finally, we investigate the change in prediction accuracy for varied interaction strengths between electrons in the system and find that there is an intermediate range in which trained models make the worst predictions.

2. Background Theory

2.1 The Many-Body Problem

The many-body problem refers to a system in which the quasi-particles involved move under the influence of the interaction force between them. For example, electrons in a box interacting by their Coulomb repulsion or even classically, the three-body problem describing how three bodies move in space by the gravitational forces between each other [14]. This extra degree of complexity results in unpredictable behaviour of the particles in the system, making this class of problem extremely difficult, if not impossible to solve analytically or numerically.

For quantum systems, the number of dimensions of the Hilbert space needed to describe the wavefunction of the many-body system scales exponentially with the total number of particles. As we will see later for a lattice model, significant issues arise in any numerically exact solutions of the quantum many-body problem, due to limitations in computer RAM needed to store very large quantum states. The problem then turns to finding ways of reducing the dimensionality of the system and extracting only the most important features of the wavefunction in order to bypass the exponentially scaling Hilbert space and allow the behaviour of larger many-body systems to be investigated [1, 9].

2.2 The Spinless Hubbard Model

2.2.1 Second Quantisation

The Hubbard model is a Hamiltonian first introduced by John Hubbard [4] to model electronic correlations in narrow energy bands and progressed into the treatment of a system of interacting electrons in a solid [15].

The dynamics of N_e electrons moving in a lattice of static ions is governed by the Hamiltonian

$$H = \sum_{i=1}^{N_e} \left(\frac{\boldsymbol{p}_i^2}{2m} + U(\boldsymbol{r}) \right) + \frac{1}{2} \sum_{i,j \neq j}^{N_e} V\left(\boldsymbol{r}_i - \boldsymbol{r}_j\right) \tag{2.1}$$

where $U(\boldsymbol{r})$ is the periodic potential of the ions and $V(\boldsymbol{r}) = \frac{e^2}{|\boldsymbol{r}|}$ is the Coulomb repulsion between electrons.

We want to move from this classical Hamiltonian to quantum mechanics by second quantization in a suitable basis of states. After first quantisation, eq. 2.1 in terms of fermionic wavefunction $\psi(\boldsymbol{r})$ becomes

$$H = \int d^3\boldsymbol{r}\psi^\dagger(\boldsymbol{r}) \left(-\frac{\hbar^2\nabla^2}{2m} + U(\boldsymbol{r})\right)\psi(\boldsymbol{r}) + \frac{1}{2}\int d^3\boldsymbol{r}d^3\boldsymbol{r}'\psi^\dagger(\boldsymbol{r})\psi^\dagger\left(\boldsymbol{r}'\right)V\left(\boldsymbol{r}-\boldsymbol{r}'\right)\psi(\boldsymbol{r})\psi\left(\boldsymbol{r}'\right) \tag{2.2}$$

Second quantisation of the above equation is achieved by promoting the dynamical quantities $\psi(\boldsymbol{r})$ and $\psi^\dagger(\boldsymbol{r})$ to operators through the anti-commutation relations describing fermions

$$\begin{aligned} \left\{\psi(\boldsymbol{r}), \psi^\dagger\left(\boldsymbol{r}'\right)\right\} &= \delta\left(\boldsymbol{r}-\boldsymbol{r}'\right) \\ \left\{\psi^\dagger(\boldsymbol{r}), \psi^\dagger\left(\boldsymbol{r}'\right)\right\} &= \left\{\psi(\boldsymbol{r}), \psi\left(\boldsymbol{r}'\right)\right\} = 0 \end{aligned} \tag{2.3}$$

We can express $\psi(\boldsymbol{r})$ as a linear combination of localised atomic orbitals $\phi_\mu(\boldsymbol{r})$ with operator coefficients $\hat{\mathrm{c}}_\mu$, where $\phi_{\mathrm{i}}(\boldsymbol{r}) = \phi\left(\boldsymbol{r}-\boldsymbol{R}_i\right)$ is a hydrogen 1 s orbital centred at lattice site position $\boldsymbol{R}_i$.

$$\psi(\boldsymbol{r}) = \sum_\mu \hat{c}_\mu\phi_\mu(\boldsymbol{r}), \quad \psi^\dagger(\boldsymbol{r}) = \sum_\mu \hat{c}_\mu^\dagger\phi_\mu^*(\boldsymbol{r}) \tag{2.4}$$

Substituting this expansion for $\psi(\boldsymbol{r})$ into the first quantised Hamiltonian (eq. 2.2), we arrive at the second quantised form

$$\widehat{H} = \sum_{ij} H_{ij}\hat{c}_i^\dagger\hat{\mathrm{c}}_j + \frac{1}{2}\sum_{ijkl} V_{ijkl}\hat{c}_i^\dagger\hat{c}_j^\dagger\widehat{\mathrm{c}}_k\hat{\mathrm{c}}_l \tag{2.5}$$

which is the sum of the single-particle Hamiltonian (quadratic in c operators) and interaction Hamiltonian (quartic in c operators) with

$$H_{ij} = \int d^3\boldsymbol{r}\phi_i^*(\boldsymbol{r})\left(-\frac{\hbar^2\nabla^2}{2m} + U(\boldsymbol{r})\right)\phi_j(\boldsymbol{r}) \tag{2.6}$$

$$V_{ijkl} = \frac{1}{2}\int d^3\boldsymbol{r}d^3\boldsymbol{r}'\phi_i^*(\boldsymbol{r})\phi_l(\boldsymbol{r})V\left(\boldsymbol{r}-\boldsymbol{r}'\right)\phi_j^*\left(\boldsymbol{r}'\right)\phi_k(\boldsymbol{r}) \tag{2.7}$$

We now make several approximations to simplify the model. Firstly, in this research

project, we are dealing with the spinless-Hubbard mode, which means each lattice site is either occupied or unoccupied $\{0, 1\}$. Therefore, the interaction parameter V_{ijkl} gives its largest contribution for occupied neighbouring sites ($i = l, j = k = i + 1$), denoted V, and can be neglected elsewhere. We also assume a nearest-neighbour tight-binding approximation, where we only keep hopping matrix elements between nearest neighbours [15]. Introducing on-site energies $\varepsilon_i = H_{ii}$, hopping parameter $t = -H_{i,i+1} = -H_{i+1,i}$, and particle number operators $\hat{n}_i = \hat{c}_i^\dagger \hat{c}_i$, we finally arrive at the spinless Hubbard model for a one-dimensional chain of L atoms

$$\widehat{H}_U = \widehat{U} + \widehat{T} + \widehat{V} = \sum_{i=1}^{L} \varepsilon_i \hat{n}_i - t \sum_{i=1}^{L} \left(\hat{c}_i^\dagger \hat{c}_{i+1} + \hat{c}_{i+1}^\dagger \hat{c}_i \right) + V \sum_{i=1}^{L} \hat{n}_{i+1} \hat{n}_i \tag{2.8}$$

Note that the Hubbard model is essentially a one parameter model, depending only on the ratio V/t. In this report, we fix $t = 1$, setting the energy scale of the system, and vary V freely to adjust its properties. We can interpret the Hamiltonian by considering the action of each term. The creation operator $\left(\hat{c}_i^\dagger\right)$ and annihilation operator $(\hat{c}_i)$ have anti-commutation relations following from eq. 2.3

$$\begin{aligned} \left\{\hat{c}_\mu^\dagger, \hat{c}_\nu\right\} &= \delta_{\mu\nu} \\ \{\hat{c}_\mu, \hat{c}_\nu\} = \left\{\hat{c}_\mu^\dagger, \hat{c}_v^\dagger\right\} &= 0 \end{aligned} \tag{2.9}$$

Therefore, they act on occupied and unoccupied lattice sites as follows:

$$\begin{aligned} \hat{c}^\dagger|0\rangle = |1\rangle, \quad \hat{c}^\dagger|1\rangle = 0 \\ \hat{c}|0\rangle = 0, \quad \hat{c}|1\rangle = |0\rangle \end{aligned} \tag{2.10}$$

The first term $\widehat{U} = \sum_i \varepsilon_i \hat{c}_i^\dagger \hat{c}_i$ destroys an electron at site i and recreates it at the same site. Therefore $|0\rangle$ and $|1\rangle$ are eigenvectors of the number operator $\hat{n}_i$ with eigenvalues 0 and 1 , respectively. So, the first term describes the interaction between the electron and nuclei by assigning an occupied state i an energy ε_i. The kinetic term $\widehat{T}$ destroys an electron at site i and recreates it at site $i + 1$, or vice versa, with an energy cost of such a transfer being t. Finally, the Coulomb interaction term $\widehat{V}$ is only non-zero if sites i and $i + 1$ are both occupied, requiring an additional energy V for such an arrangement [16].

2.2.2 Exact Diagonalisation

A common method for numerically solving the Hubbard Hamiltonian (eq. 2.8) is called exact diagonalisation. It involves calculating the matrix elements of $\widehat{H}_U$ in a convenient

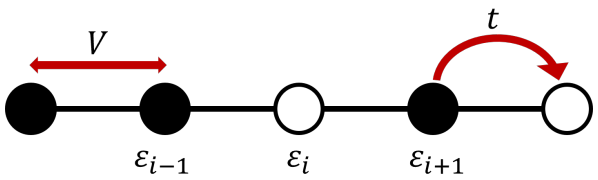

Figure 2.1: The spinless Hubbard model for a lattice chain. Each site can either be occupied (black) or unoccupied (white). Electrons can hop from an occupied state to the nearest unoccupied neighbour with a transition energy t. There is an interaction energy V for neighbouring occupied states.

basis of the Hilbert space of the system, and diagonalise the resulting matrix [17]. For this research project, we focused on the case of a lattice chain of four lattice sites and two electrons. The basis vectors of the Hilbert space describing this system are given by all the ways of occupying four sites with two electrons, namely

$$\begin{aligned} |1\rangle = |1,1,0,0\rangle, \quad |2\rangle = |1,0,1,0\rangle, \quad |3\rangle = |1,0,0,1\rangle, \\ |4\rangle = |0,1,1,0\rangle, \quad |5\rangle = |0,1,0,1\rangle, \quad |6\rangle = |0,0,1,1\rangle \end{aligned} \tag{2.11}$$

A wavefunction describing the system can then be written as a linear combination over this basis set

$$|\psi\rangle = \sum_{i}^{dim\mathcal{H}} \psi_i |i\rangle \tag{2.12}$$

Finding the Hubbard Hamiltonian describing the system in a matrix representation over the basis set is simply found by calculating the individual matrix elements $H = \left(\widehat{H}_U\right)_{ij} = \left\langle i \middle| \widehat{H}_U \middle| j \right\rangle$. The atomic potential terms $(\widehat{U})_{ij}$ and interaction terms $(\widehat{V})_{ij}$ are diagonal entries in the Hamiltonian matrix, where the former counts the number of electrons at each site and applies its associated energy, and the latter adds an energy contribution of V for every pair of occupied neighbouring sites in the basis vector. The kinetic term $(\widehat{T})_{ij}$ adds an entry to H if another basis state can be created by transferring one electron. For example, $\widehat{T}|5\rangle = -t(|3\rangle + |4\rangle + |6\rangle)$, and so $(\widehat{T})_{3,5} = \langle 3|\widehat{T}|5\rangle = -t, (\widehat{T})_{2,5} = \langle 2|\widehat{T}|5\rangle = 0$. Adding up all the energy contributions for each term in $\widehat{H}_U$, we arrive at the following matrix representation of $\widehat{H}_U$

$$H = \begin{pmatrix} \varepsilon_1+\varepsilon_2+V & -t & 0 & 0 & 0 & 0 \\ -t & \varepsilon_1+\varepsilon_3 & -t & -t & 0 & 0 \\ 0 & -t & \varepsilon_1+\varepsilon_4 & 0 & -t & 0 \\ 0 & -t & 0 & \varepsilon_2+\varepsilon_3+V & -t & 0 \\ 0 & 0 & -t & -t & \varepsilon_2+\varepsilon_4 & -t \\ 0 & 0 & 0 & 0 & -t & \varepsilon_3+\varepsilon_4+V \end{pmatrix}$$

The Schrodinger equation $H\boldsymbol{\psi} = E\boldsymbol{\psi}$ takes the form of an eigenvalue problem whose solutions are obtained by diagonalising H above, yielding the energy eigenvalues E^n and corresponding eigenstates $\boldsymbol{\psi}^n = (\psi_1^n, \ldots, \psi_6^n)$ of the Hubbard Hamiltonian (eq. 2.8). The corresponding wavefunctions are therefore $|\psi^n\rangle = \sum \psi_i^n |i\rangle$.

More generally, the dimension of the Hilbert space N_H for a system of L lattice sites and N_e electrons is given by the number of ways of distributing N_e electrons over L sites, namely

$$N_H = \begin{pmatrix} L \\ N_e \end{pmatrix} = \frac{L!}{N_e!\,(L-N_e)!} \tag{2.13}$$

We see a severe exponential scaling of the Hilbert space with the number of particles in the system for a many-body problem, e.g., $N_H = 184,756$ for a system of 20 sites with half-filling. Hence, computations are limited to systems with a few tens of particles due to the increased computer memory required to store $N_H \times N_H$ matrix and a diagonalisation whose computational time scales as N_H^3. For larger systems, other methods of diagonalisation which are less computationally expensive need to be employed, such as Lanczos algorithm [18] or storing matrices in sparse matrix format.

2.3 Density Functional Theory

Density functional theory provides an alternative method to solving a many-body problem which avoids calculating the many-body wavefunction, which we've seen in section 2.2.2 to scale exponentially with system size. We can alternatively examine the system's properties using the fundamental electron density describing the system, a function of three spatial coordinates.

The foundation for density functional theory is based on the Hohenberg-Kohn (HK) theorems [2]. According to the first theorem, for a system of electrons moving under the influence of an external potential $v(\boldsymbol{r})$ and their mutual Coulomb repulsion (eq. 2.1), there exists a unique functional between $v(\boldsymbol{r})$ and the electron density $n(\boldsymbol{r})$ in the

ground state. Since $v(\boldsymbol{r})$ determines the Hamiltonian, the total energy E is a unique functional of $n(\boldsymbol{r})$, given by

$$E[n(\boldsymbol{r})] = \int n(\boldsymbol{r})v(\boldsymbol{r})d\boldsymbol{r} + F[n(\boldsymbol{r})] \tag{2.14}$$

where $F[n(\boldsymbol{r})]$ is a universal functional independent of $v(\boldsymbol{r})$. The second theorem states that at the ground state electron density $E[n(\boldsymbol{r})]$ is minimised, with the restriction $N[n] = \int n(\boldsymbol{r})d\boldsymbol{r} = N$ is conserved.

$$E_{GS} = E\left[n_{GS}(\boldsymbol{r})\right] < E[n(r)] \tag{2.15}$$

As a result, the ground state electron density specifies the Hamiltonian of the system and hence its solutions. We can provide an analogous formulation for the HK theorems on an interacting lattice model, such as the Hubbard model (eq. 2.8) [19]. In this case, the density describing the system is now the average electron occupation of each lattice site given by

$$n_i = \left\langle \psi \left| \hat{c}_i^\dagger \hat{c}_i \right| \psi \right\rangle \tag{2.16}$$

Lattice DFT states the one-to-one correspondence between the site occupation $\boldsymbol{n}$ and the external potential, defined by the on-site energies ε_i. Therefore, the Hohenberg-Kohn theorems in the lattice formulation are given by

$$E[\boldsymbol{n}] = \sum_i n_i\varepsilon_i + F[\boldsymbol{n}], \quad E_{GS} = E\left[\boldsymbol{n}_{GS}\right] < E[\boldsymbol{n}] \tag{2.17}$$

As a consequence, the many-body wavefunction is a functional of the average site occupation; $|\psi\rangle = |\psi[\boldsymbol{n}]\rangle$, meaning the system can be studied using a quantity that depends linearly on system size, rather than the exponentially expanding wavefunction.

A density functional formalism has also been developed for time-dependent systems [5]. Therefore, we can study the time evolution of quantum systems through the time evolution of the appropriate electron density describing the system. The problem with DFT remains that the universal functional F is unknown and needs to be approximated. Machine learning has proved to be a viable approach to constructing exact functionals for the Hubbard model [12] and tackling time-dependent systems [10]. In this research, we focus on using a machine learning approach to approximate the site occupation time propagation for the spinless Hubbard model.

2.4 Time Evolution of Quantum Systems

The density matrix is an alternative way of representing the quantum state of a system, in that it contains the minimum set of information necessary to calculate the measurable quantity of any observable of a system prepared according to given specifications [20]. For a system prepared in a pure state $|\psi\rangle$, the density matrix is given by $\hat{\rho} = |\psi\rangle\langle\psi|$. For a system in a mixed state, given by the superposition of pure states $|\psi_i\rangle$ weighted by probability p_i, the density matrix is expressed as $\hat{\rho} = p_i \, |\psi_i\rangle \, \langle\psi_i|$.

The density matrix is used to find the expectation value of an operator $\hat{A}$ as

$$A = \langle \hat{A} \rangle = \sum_i p_i \left\langle \psi_i | \hat{A} | \psi_i \right\rangle = \mathrm{Tr}(\hat{\rho}\hat{A}) \tag{2.18}$$

Investigating how the density matrix describing a system evolves with time, we first note that any unitary transformation U that changes $|\psi\rangle$ representing a pure state into $U|\psi\rangle$ also changes $\hat{\rho}$ representing that state into $U\hat{\rho}U^\dagger$ [20]. The time dependence of the wavefunction is controlled by the time-dependent Schrödinger equation, given by

$$i\hbar \frac{\partial}{\partial t} |\psi(t)\rangle = \widehat{H}\psi(t) \tag{2.19}$$

It is solved by the unitary evolution with the Hermitian operator $U(t) = e^{-i\widehat{H}t/\hbar}$, such that $|\psi(t)\rangle = e^{-i\widehat{H}t/\hbar}|\psi(0)\rangle$. Then, the time evolution of the density operator can simply be found by taking the time derivative of $\widehat{\rho}$ and substituting eq. 2.19 to arrive at a first-order differential equation called the Liouville von Neumann equation,

$$\frac{\partial \hat{\rho}}{\partial t} = -\frac{i}{\hbar}[\widehat{H}, \hat{\rho}] \tag{2.20}$$

which has the solution $\hat{\rho}(t) = e^{-i\widehat{H}t/\hbar}\hat{\rho}(0)e^{i\widehat{H}t/\hbar}$. Then we have that the expectation value of observable $\hat{A}$ varies in time according to

$$A(t) = \mathrm{Tr}[\hat{\rho}(t)\hat{A}] \tag{2.21}$$

Then the site occupation evolution can be explicitly calculated as $n_i = \mathrm{Tr}\,[\hat{\rho}(t)\hat{n}_i]$.

One also notes that the expectation value of any operator that commutes with the system's Hamiltonian remains constant in time. For the Hubbard Hamiltonian, $\widehat{H}_U$ commutes with itself and the total number operator $\widehat{N} = \sum_i \hat{n}_i$, therefore the total energy and number of particles of the system remains constant with time [21].

3. Computational Methods

3.1 Machine Learning

Machine learning is a computational technique, modelled on the operation of the brain, which is exposed to sets of training data, with the goal of making accurate predictions on unseen data. Machine learning, particularly neural networks, has proven to be very effective in a variety of fields, ranging from image and speech recognition [8] to industrial [6] and medical problems [7]. It has now seen wide use in the field of condensed matter physics. Given that the many-body problem is a notoriously difficult problem to solve analytically paired with the ability to the ability to generate large datasets, it makes it a good candidate to be studied using machine learning techniques. In this project, machine learning is used to approximate the propagation of a many-body systems in time.

3.1.1 Neural Networks

The fundamental unit of a neural network is called the perceptron (fig 3.1), based on the McCulloch and Pitts model of a biological neuron [22]. The concept of 'learning' using the perceptron is based on least-squares regression and logistic regression [23].

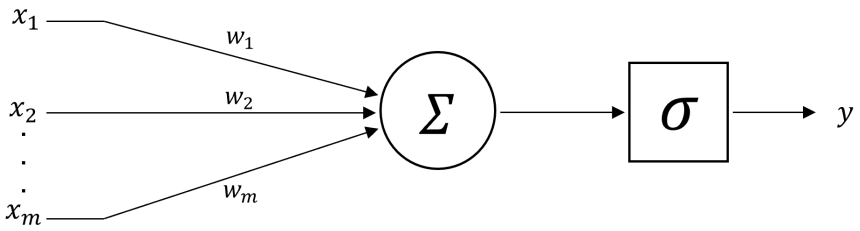

Figure 3.1: The perceptron - basic unit of a neural network. Inputs in the feature vector $\mathbf{x}$ are combined with a set of weights $\{w_i\}$ and passed through an activation function σ outputting prediction $\hat{y}$

The model aims is to 'learn' the mapping $f : \boldsymbol{x} \rightarrow \boldsymbol{y}$ between a collection of inputs $\boldsymbol{x} = (x_1, \ldots, x_n)$ and outputs $\boldsymbol{y} = (y_1, \ldots, y_N)$ using a set of adjustable weights $\boldsymbol{w} = (w_1, \ldots, w_n)$. The procedure is divided into three main parts. Firstly, the feature vector $\boldsymbol{x}$ combines with the set of weights $\boldsymbol{w}$, offset with a bias b, which can be treated as another weight that combines with a unit input.

$$a = \sum_{i=1}^{n} x_i w_i + b$$

The sum is then passed through an activation function, which outputs a prediction $\hat{y} = \sigma(a)$. The activation function is chosen depending on the application of the model. Finally, we define a loss function, which quantifies the error in the prediction made by the model for the given set of weight parameters. This research project deals with regression predictions, for which a linear activation function with a mean square error loss function (eq. 3.1) is an appropriate choice.

$$\mathcal{L}_{MSE}(\boldsymbol{w}) = \frac{1}{N} \sum_{k=1}^{N} (y_i - \hat{y}_i(\boldsymbol{w}))^2 \tag{3.1}$$

The model learns the mapping f between inputs and outputs by feeding the network the training data in cycles called epochs and adjusting the weights $\{w_i\}$ accordingly until $\mathcal{L}_{MSE}(\boldsymbol{w})$ converges to a minimum. Once the parameters $\{w_i\}$ are fixed, the network can make predictions on unseen inputs.

To make more powerful models capable of learning non-trivial patterns and correlations in the mapping function between inputs and outputs, perceptron units are joined together in multiple layers, where units in adjacent layers are fully connected by another set of adjustable weights (fig. 3.2). Multilayer networks consist of three components; an input layer, one or more hidden layers whose computations are not visible to the user, and an output layer giving the model's prediction.

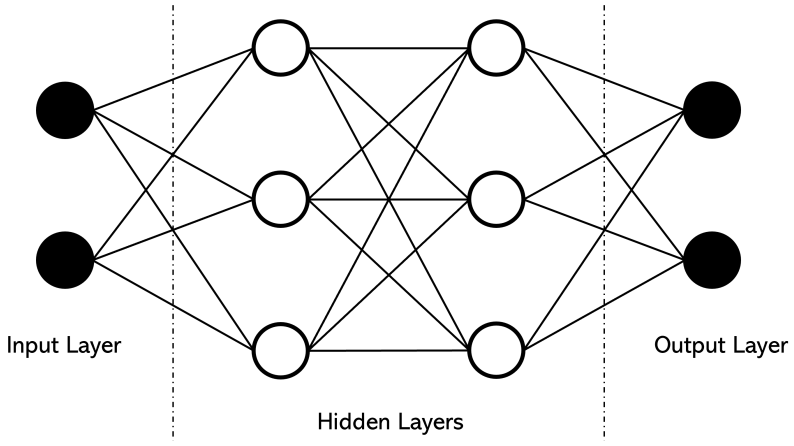

Figure 3.2: Full connected neural network. The circles represent a perceptron unit (fig. 3.1) which are all connected by a set of weights. The aim is to adjust the weights until the error in the prediction of the output layer is minimised.

A linear activation function is only effective up to one layer deep. By taking the activation function of the hidden neurons to be non-linear, the network can create

highly optimised composition functions [23]. Such a network is very efficient at learning non-linear mappings between sets of variables. A common choice is the sigmoid activation function $\sigma(a) = \frac{1}{1+e^{-a}}$ or ReLU which is a common replacement in modern neural networks [24].

A multilayer network arrives as an output from an input by the process of forward propagation through each layer. The output of the l^{th} layer in the network, denoted $\mathbf{z}^{(l)}$, is found by combining the input from the previous layer $\mathbf{z}^{(l-1)}$ with the weight matrix and bias vector associated with the previous layer, $W^{(l-1)}$ and $\boldsymbol{b}^{(l-1)}$ respectively, passed through an activation function σ.

$$\mathbf{z}^{(l)} = \sigma\left(\boldsymbol{a}^{(l-1)}\right) = \sigma\left(W^{(l-1)}\mathbf{z}^{(l-1)} + \boldsymbol{b}^{(l-1)}\right) \tag{3.2}$$

The output of the final layer gives the prediction of the model. Training the network, once again, entails determining the set of weights and biases which minimises the loss function. However, in the case of a multilayer network, the loss function becomes a complex surface over the weight space, whose minimisation is extremely non-trivial [23]. We use an iterative process called backpropagation [25] to perform such minimisation. Given the mean square error loss function (eq. 3.1), using the chain rule of differential calculus, the derivative of the loss function with respect to a single weight w_{ij}^{l} in the l^{th} layer is given by the product of known derivatives

$$\frac{\partial \mathcal{L}}{\partial w_{ij}^{(l)}} = \frac{\partial z_i^{(l)}}{\partial w_{ij}^{(l)}} \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}} \frac{\partial \mathcal{L}}{\partial a_i^{(l)}} \tag{3.3}$$

The total gradient of each layer for every neuron j is given by the sum of eq. 3.3 over all the connections i emerging from neuron j. The weights are updated by gradient descent.

$$\boldsymbol{w}_j^{(l)} \rightarrow \boldsymbol{w}_j^{(l)} - \nabla_{w_j}\mathcal{L} \tag{3.4}$$

The process of gradient descent continues until the loss function converges. One has to consider the possibility of the network overfitting the data. Overfitting is the case in which the neural network learns the patterns and behaviour of the training data very well, but misses the underlying trend in the entire dataset, producing relatively large errors in the unseen test data. A common method for mitigating the effect of overfitting is early stopping [26]. The process involves computing an error in the prediction on a test set of data throughout the process of training. This is achieved by splitting our initial dataset into three sections; a training set, a validation set, and a test set. The

validation set is used alongside the training set computing a validation error after each training epoch. Gradient descent is halted when the validation error begins to rise, giving a model ready to make predictions on the test set.

3.1.2 Data Generation for Machine Learning

For the system of four sites and two electrons, whose time evolution we are attempting to predict, we require data for its exact time evolution from a sample of input states in order to predict the evolution of new input states. In the density matrix description of the system, the evolution is determined by the Hamiltonian H and initial state ρ_0 by the equation $\rho_t = e^{-iHt/\hbar}\rho_0 e^{iHt/\hbar}$. Therefore, given a uniform set of N_s initial states $\boldsymbol{\psi}_0$ in the basis 2.11, a training dataset is simply created by calculating ρ_t at N_t timesteps along its evolution. So, our training dataset will have the shape (N_S, N_t, N_H, N_H), where $N_H = 6$ is the Hilbert space dimension.

The initial state $|\psi_0\rangle \mapsto \boldsymbol{\psi}_0 = (\psi_1, \psi_2, \ldots, \psi_6)^T$ in the basis 2.11 obeys the normalisation condition $\boldsymbol{\psi}_0^\dagger \boldsymbol{\psi}_0 = \sum_{i=1}^6 |\psi_i|^2 = 1$, therefore $\boldsymbol{\psi}_0$ is any point lying on the surface of a six-dimensional unit hypersphere. A uniform distribution of data in the training set is imperative to ensuring a model that accurately represents its purpose and makes precise predictions on unseen test sets. A nonuniform distribution of states $\boldsymbol{\psi}_0$ on the 6-sphere will result in biases in the data towards certain states and inaccurately representing the underlying behaviour in the overall set of possible states [27].

A uniform distribution over the 6-sphere is achieved by sampling each coordinate ψ_i from the normal distribution. A point x sampled from the normal distribution has a probability distribution $g(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2}$. Therefore, the probability distribution for $\boldsymbol{\psi}_0$ whose elements are sampled from a normal distribution is

$$g(\boldsymbol{\psi}_0) = g(\psi_1) \cdots g(\psi_6) = \left(\frac{1}{\sqrt{2\pi}} e^{-\psi_1^2}\right) \cdots \left(\frac{1}{\sqrt{2\pi}} e^{-\psi_6^2}\right) = \frac{1}{(2\pi)^3} e^{-\sum_{i=1}^6 \psi_i^2} = \frac{e^{-1}}{(2\pi)^3}$$

where $\sum_{i=1}^6 \psi_i^2 = 1$ by the normalisation condition. Hence, $g(\boldsymbol{\psi}_0)$ is constant and the state $\boldsymbol{\psi}_0$ has an equal probability of residing at any point on the 6 -sphere. In figure 3.3, we plot a number of normalised three-dimensional vectors $v = (x, y, z)$ for two separate coordinate samplings. The first case is where polar coordinated $\{\theta, \phi\}$ are chosen randomly in their respective ranges $0 \leq \theta < 2\pi$ and $0 \leq \phi < \pi$. We observe an accumulation of points at certain regions of the sphere due to non-uniform dependence of the probability distribution on θ and ϕ. In the other case, $\{x, y, z\}$ are sampled from the normal distribution which has a uniform distribution over the surface of the sphere, as reflected in figure 3.3. This will provide an appropriately uniform dataset for

machine learning [28].

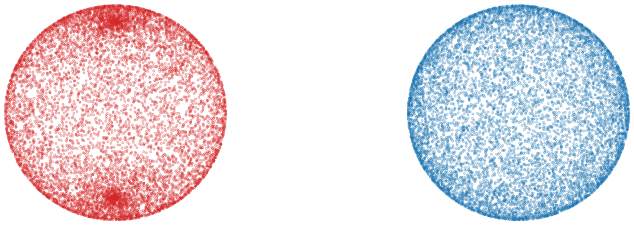

Figure 3.3: Sampling random points on the unit sphere. Left: Polar coordinates θ, ϕ are chosen randomly, resulting in a non-uniform sample of states. Right: The Cartesian coordinates x, y, z are sampled from the Gaussian distribution resulting in a suitably uniform dataset for machine learning.

3.2 Learning Quantum Time Evolution

In section 2.4, we showed the evolution of the density matrix obeys the Liouville von Neumann equation (eq. 2.20). As a result, for discrete timesteps, the future timestep of the density matrix only depends on its current state. Given the first-order ODE $\frac{\partial \rho}{\partial t} = -\frac{i}{\hbar}[\widehat{H}, \rho] = f(\rho, t)$, using the derivative from first principles, we find

$$\frac{\rho(t + \Delta t) - \rho(t)}{\Delta t} = f(\rho, t) \Leftrightarrow \rho_{t+1} = \rho_t + \Delta t f\left(\rho_t\right) \tag{3.5}$$

Hence, a model trained on systems in the density matrix formalism should only require the current timestep to learn the mapping to the next timestep, $\tilde{\rho}_{t+1} = F\left(\rho_t\right)$. One can predict subsequent timesteps using autoregression, where the output of the model at one timestep is used as the input to predict the next timestep [30]. This technique allows an entire time series to be reconstructed from an initial input by

$$\tilde{\rho}_{t+m} = \underbrace{F(F(\ldots F}_{m}\left(\rho_t\right)\ldots)) \tag{3.6}$$

For predicting large timesteps into the future, autoregression can be slow and cumulative errors decrease the accuracy in the prediction. To remedy this, one can train models learning the mapping F_m to various timesteps ahead by changing the target output of the training dataset.

$$\tilde{\rho}_{t+m} = F_m\left(\rho_t\right) \tag{3.7}$$

Training a select number of models (e.g. $F_1, F_2, F_5, F_{10}, F_{20}, \dots$), one can access any timestep into the future with very few autoregression steps, bypassing the cumulative autoregression error.

The same processes are carried out for learning the site occupation evolution; however, one finds it is no longer sufficient to use the current timestep only to make accurate predictions for the next and future timesteps in this reduced representation of the system. The model will require a collection of N_{in} input timesteps of the system's history, now learning the mapping

$$\widetilde{\boldsymbol{n}}_{m+1} = F\left(\{\boldsymbol{n}_m, \boldsymbol{n}_{m-1}, \dots, \boldsymbol{n}_{m+N_{in}}\}\right) \tag{3.8}$$

Autoregression is carried out in the same way to predict the time series over a time interval, except here we cycle out the input site occupation vectors with the predicted ones, e.g., $\widetilde{\boldsymbol{n}}_{m+2} = F\left(\{\widetilde{\boldsymbol{n}}_{m+1}, \boldsymbol{n}_m, \boldsymbol{n}_{m-1}, \dots, \boldsymbol{n}_{m+N_{in}+1}\}\right)$.

The implementation of the above networks is executed using Keras [31], which carries out the computation of the loss function, its optimisation and gradient descent. For all the models used in this project, we generated the entire dataset for 2000 uniform initial samples and propagated its exact evolution over 500 timesteps. The (train: validate: test) ratio was set to (70: 20: 10). For the density matrix models, it was found that a linear model was sufficient in achieving validation errors $\sim 10^{-15}$. For the site occupation models, we used a network of one hidden layer of 32 neurons with the ReLU activation function, which was found to give the smallest validation loss.

4. Results and Discussion

4.1 Learning Density Matrix Time Evolution

Figure 4.1 shows the results for the model trained on a set of time-evolved density matrices, aiming to predict its next timestep for an interaction strength $V = 4$ and timestep length $\Delta t = 0.04/J$, where $\hbar = 1$ in $U(t) = e^{-iHt/\hbar}$. Choosing a random sample from a test set not previously seen by the model, the exact evolution of the average occupations for the four lattice sites was calculated using $n_{it} = \mathrm{Tr}\left[\rho_t \hat{n}_i\right]$, where the density matrix evolution ρ_t is governed by the solution to the Von-Neumann equation 2.20. Using the same initial state, the predicted time evolution was found using the trained model by autoregression. Calculating the average occupation from the predicted density matrices at each timestep using $\tilde{n}_{it} = \mathrm{Tr}\left[\tilde{\rho}_t \hat{n}_i\right]$, we provide a comparison to the exact numerical solution in fig. 4.1.

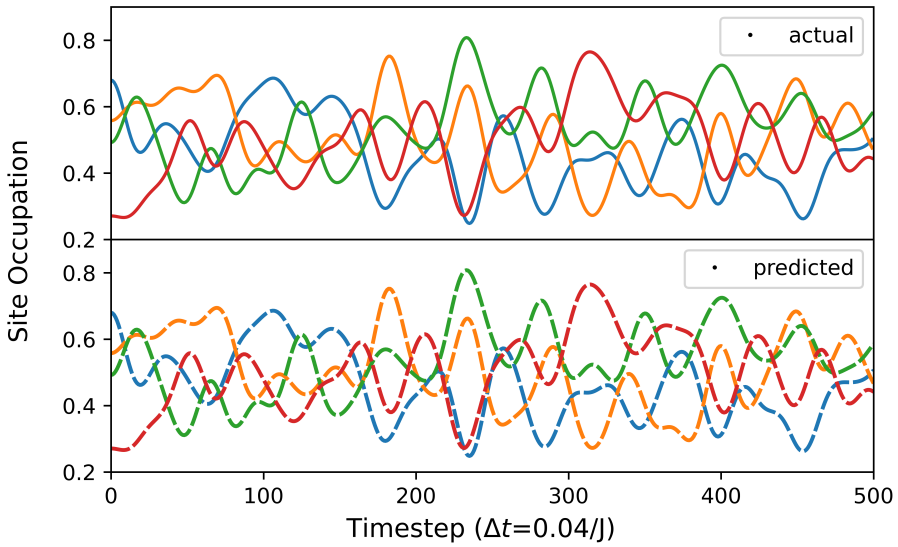

Figure 4.1: Time evolution of the average occupations of the four lattice sites [n_1 (blue), n_2 (orange), n_3 (red), n_4 (green)]. Top: Numerical solution governed by the solution to eq. 2.20. Bottom: Evolution predicted by our model given the same initial state.

The mean absolute error for the independent density matrix components over the entire time series was found to be 8.62×10^{-6}, averaged over 100 sample time series in the test set. Hence, the model trained on one input density matrix is capable of accurately

predicting the system's evolution, demonstrating that the current state of the system in the density matrix formalism includes sufficient information in predicting the next and subsequent timesteps in the time series, as expected from eq. 3.5.

Figure 4.2 shows the growth of the error with time for autoregressive density matrix predictions given the test set of 100 initial states ρ_0. In the top panel, the mean absolute error between the actual density matrix (ρ) and predicted density matrix ($\tilde{\rho}$) at each timestep in the series is calculated by $\Delta_t = \frac{1}{n}\sum_{i=1}^{n} |\rho_t^i - \tilde{\rho}_t^i|$ where $n = 36$ is the number of independent (real and complex) components of the density matrix. In the bottom panel, the error in the predicted energy $\tilde{E}_t = \mathrm{Tr}[\tilde{\rho}_t H]$ is also plotted.

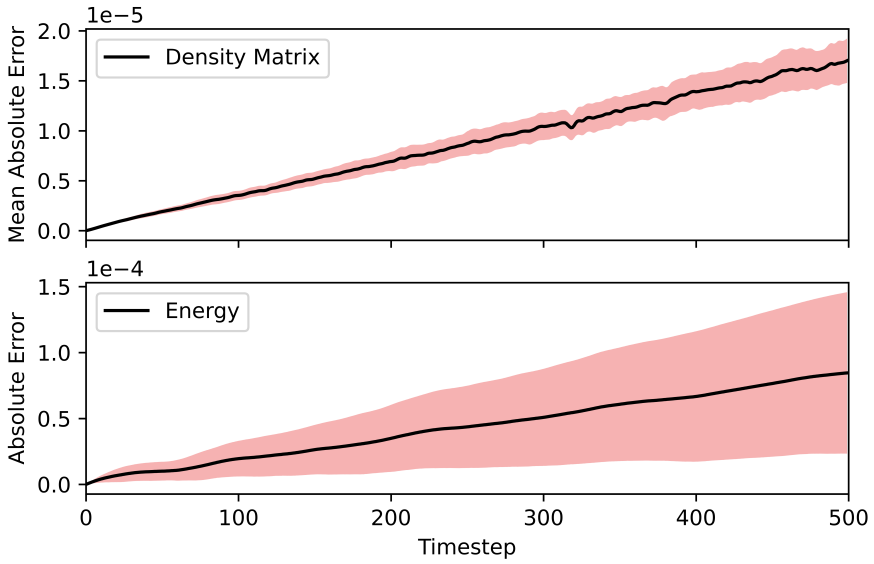

Figure 4.2: Growth in the error of the predicted system evolution using autoregression. Top: Mean absolute error of the independent density matrix elements. Bottom: Absolute error in the system's energy (The average energy of the 100 sample systems was $E_{avg} = 1.88J$). The black line indicates the mean error over 100 test sample time series and the shaded red region indicates the standard deviation of the prediction error.

One can avoid the accumulative error of autoregressive predictions by training a select number of models predicting varying numbers of timesteps into the future, specifically we trained models predicting $1, 2, 5, 10, 20, 50, 100$ timesteps ahead. Taking the 132^{nd} timestep for example, the error produced by 132 one-step predictions $\tilde{\rho}_{132} = F_1^{(132)}(\rho_0)$ produced an error 54 times larger than that produced by 4 autoregression steps $\tilde{\rho}_{132} = F_2(F_{10}(F_{20}(F_{100}(\rho_0))))$.

4.2 Learning Site Occupation Time Evolution

To study the evolution of the average site occupations for our system of four sites and two electrons, the current occupation vector $\boldsymbol{n}_t$ does not have enough information to

predict the next timestep $\boldsymbol{n}_{t+1}$, hence a collection of its previous history $\{\boldsymbol{n}_t, \boldsymbol{n}_{t-1}, \ldots \boldsymbol{n}_{t-N_{in}+1}\}$ is required to make accurate predictions of the system's future evolution. Figure 4.3 displays a series of predicted time series for models trained on varying numbers of input timesteps N_{in}. We plot the true and predicted evolutions for a single occupation vector component, where predicted evolutions are found by autoregression.

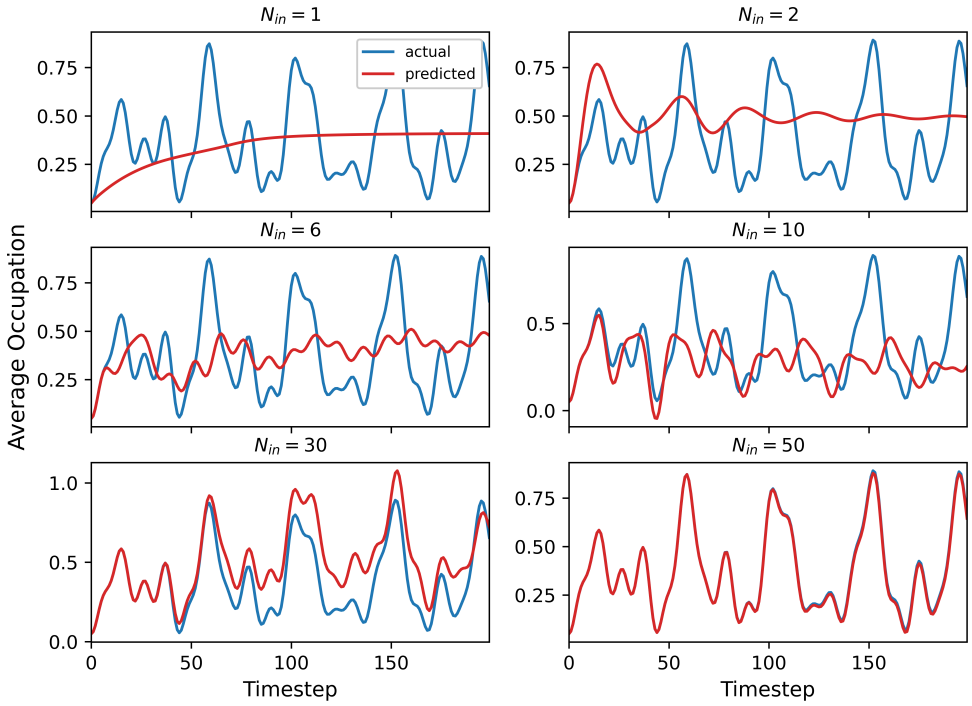

Figure 4.3: Series of plots comparing the numerically exact time evolution of a single average occupation component (blue) with that predicted by our site occupation model using autoregression (red). The predicted time series is plotted for models trained on $N_{in} = 1, 2, 6, 10, 30, 50$ input timesteps predicting the next timestep.

We find that by providing the model with an increased memory of previous timesteps, it becomes more capable of making accurate predictions of the average occupations at the next timestep, and hence, with enough input timesteps, it can make very good approximations of the system's evolution over longer time intervals. For $V = 4$, we observe the model requiring $N_{in} = 50$ previous timesteps before it can make a precise reproduction of the site occupation time series. Figure 4.4 illustrates the dependence of the accuracy of the model's prediction on the number of previous timesteps used to predict the next timestep. For each of the 100 sample time series in the test set, the mean average error over the first 20 predicted occupation vectors $\{\widetilde{\boldsymbol{n}}_{N_{in}+1}, \ldots, \widetilde{\boldsymbol{n}}_{N_{in}+20}\}$ calculated at varied N_{in} is plotted, from which we can deduce a mean dependence of N_{in} on the prediction accuracy, as shown in fig. 4.4.

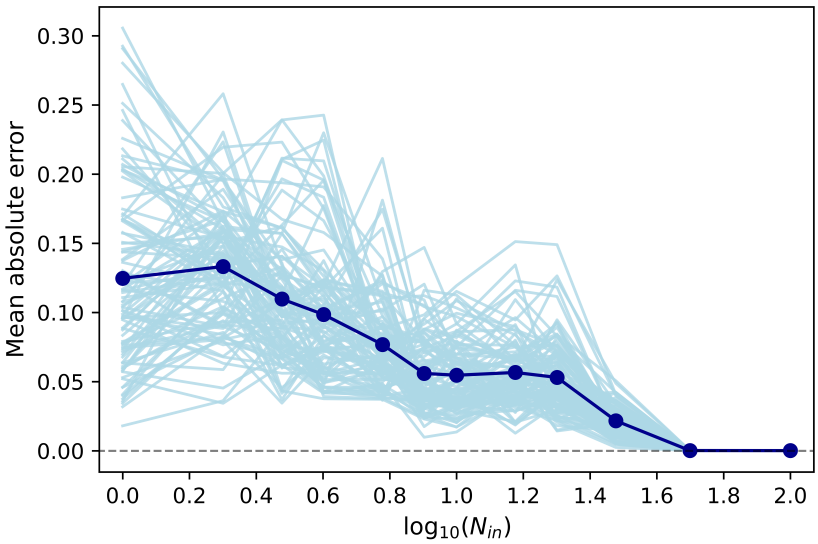

Figure 4.4: Mean absolute error of the average site occupation vector over the first 20 predicted timesteps for models trained on a varying number of input timesteps N_{in}. The dark blue line indicates the mean error of 100 sample time series (light-blue).

4.3 Varying Interaction Strengths

One of the questions we can now ask is what factors of the system influence how predictable its evolution is in the compressed representation of average occupation. Here, we investigate how varying the interaction strength V in the Hubbard Hamiltonian (eq. 2.8) affects the accuracy in the prediction. Figure 4.5 compares the mean absolute error in the first 20 predicted site occupation timesteps for the low $V = 0$, intermediate $V = 4$ (fig. 4.4), and strong $V = 20$ interaction strengths.

Figure 4.5 shows the slowest convergence in the model's prediction error for the intermediate case of $V = 4$. For the zero interacting and strong interacting cases, we observe that the model was more capable of learning the behaviour of the average site occupation evolution, making relatively accurate predictions for a history of 10 input timesteps. In figure 4.6, we keep the number of input timesteps the model is trained on fixed at $N_{in} = 10$, and calculate the mean absolute error in the site occupation predictions for varying interaction strengths to elucidate the dependence.

Figure 4.6 confirms that there is a region of intermediate interaction strength for which the behaviour of the individual site occupations is more difficult to predict and requires an increased memory of the system's history to accurately reproduce its evolution. For zero interaction strength, the model's prediction is extremely accurate, suggesting that the propagation is local in time. The region of inaccuracy peaks around $V = 2$ to $V = 6$ after which the error converges to zero for higher interaction strengths.

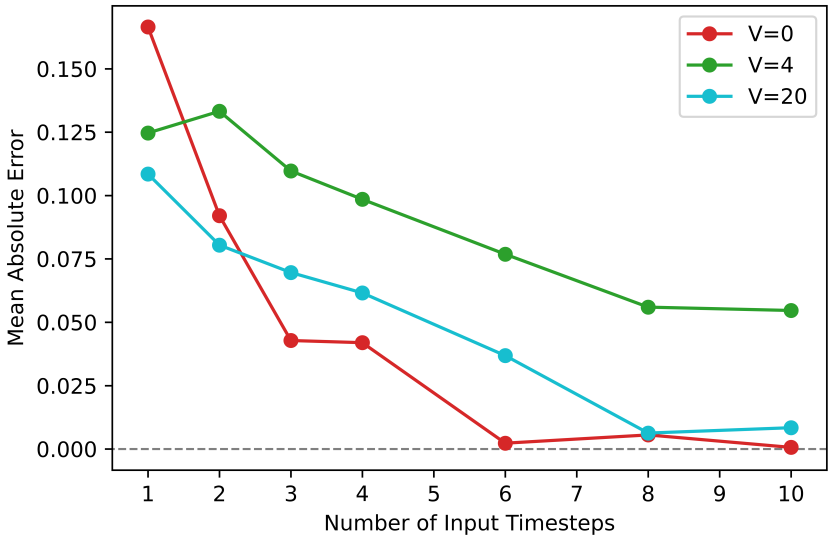

Figure 4.5: Mean absolute error of the average site occupation vector over first the 20 timesteps predicted by our model against the number of timesteps the model was trained on for different interaction strength $V = 0, 4, 20$

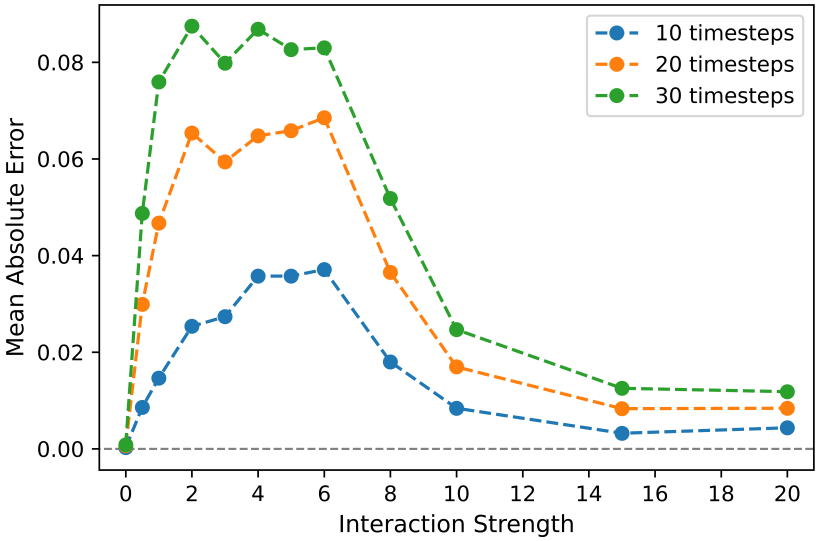

Figure 4.6: Here we illustrate how the error in our average site occupation predictions vary with interaction strength for models trained on a fixed number of 10 input timesteps. The different lines show the MAE over the first 10, 20 and 30 predicted timesteps, respectively.

5. Conclusion

In this research project, we investigated the propagation of a quantum many-body system in time using a neural network. Using the spinless Hubbard model and exact diagonalisation, we generated datasets for the exact time evolution of a quantum interacting system on a lattice, on which a neural network was trained, and made predictions of the future evolution of the system given its initial state. The Hohenberg-Kohn theorems and density functional theory provided the theoretical motivation to study the system in the reduced representation of average site occupation to bypass the exponential scaling of the Hilbert space.

In the density matrix formalism of the system, the neural network only required knowledge of the current state to make very accurate predictions for the next time step. Giving the network the initial density matrix of the system only, using autoregression, it was capable of predicting the evolution over a longer time interval to a very high precision. For a test set of 100 sample time series, the mean absolute error in the prediction over 500 time steps was computed to be 8.62×10^{-6}. The network was also capable of learning multiple time steps ahead into the future without much change in accuracy. So, by training a number of models predicting varied time steps into the future, one is capable of propagating to any specified time step into the future, avoiding the accumulation of errors by autoregression.

For the case of learning the evolution of the average site occupations of the system, the current state of the system was no longer sufficient in making accurate predictions of the next time step. The network required a collection of previous time steps to make accurate reconstructions of the site occupation time series. We then examined the effect that the interaction strength between electrons in the system had on the model's prediction. In the very low and high interaction regions, the model was able to make accurate predictions with a smaller history of time steps, whereas in the intermediate region, the behaviour of the system was more unpredictable, giving larger errors.

This research shows that machine learning coupled with density functional theory is a viable approach to learning the behaviour of a quantum many-body system, avoiding the limitations posed by determining exact numerical solutions. Potential questions stemming from this research which would be interesting to investigate is how the required memory changes with system size and the time step interval used. I would also be interested to investigate if adding the current component between neighbouring bonds reduced the required memory by any significance.

Bibliography

[1] U. Schollwöck. The density-matrix renormalization group. *Reviews of Modern Physics*, 77(1):259–315, apr 2005. doi: 10.1103/revmodphys.77.259. URL https://arxiv.org/abs/cond-mat/0409292.

[2] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136: B864–B871, Nov 1964. doi: 10.1103/PhysRev.136.B864. URL https://link.aps.org/doi/10.1103/PhysRev.136.B864.

[3] O. Gunnarsson and K. Schönhammer. Density-functional treatment of an exactly solvable semiconductor model. *Phys. Rev. Lett.*, 56:1968–1971, May 1986. doi: 10.1103/PhysRevLett.56.1968. URL https://link.aps.org/doi/10.1103/PhysRevLett.56.1968.

[4] John Hubbard. Electron correlations in narrow energy bands. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 276(1365): 238–257, 1963. doi: 10.1098/rspa.1963.0204.

[5] Erich Runge and E. K. U. Gross. Density-functional theory for time-dependent systems. *Phys. Rev. Lett.*, 52:997–1000, Mar 1984. doi: 10.1103/PhysRevLett.52.997. URL https://link.aps.org/doi/10.1103/PhysRevLett.52.997.

[6] Piyush Agarwal, Melih Tamer, M. Hossein Sahraei, and Hector Budman. Deep learning for classification of profit-based operating regions in industrial processes. *Industrial & Engineering Chemistry Research*, 59(6):2378–2395, 2020. doi: 10.1021/acs.iecr.9b04737. URL https://doi.org/10.1021/acs.iecr.9b04737.

[7] Cameron R. Olsen, Robert J. Mentz, Kevin J. Anstrom, David Page, and Priyesh A. Patel. Clinical applications of machine learning in the diagnosis, classification, and prediction of heart failure. *American Heart Journal*, 229:1–17, 2020. ISSN 0002-8703. doi: https://doi.org/10.1016/j.ahj.2020.07.009.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6): 84–90, 2017. doi: 10.1145/3065386.

[9] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. doi: 10.1126/science.1127647. URL https://www.science.org/doi/abs/10.1126/science.1127647.

[10] Irene López Gutiérrez and Christian B. Mendl. Real time evolution with neural-network quantum states. *Quantum*, 6:627, jan 2022. doi: 10.22331/q-2022-01-20-627. URL https://doi.org/10.22331%2Fq-2022-01-20-627.

[11] Anurag Anshu, Srinivasan Arunachalam, Tomotaka Kuwahara, and Mehdi Soleimanifar. Sample-efficient learning of interacting quantum systems. *Nature Physics*, 17(8):931–935, may 2021. doi: 10.1038/s41567-021-01232-0. URL https://doi.org/10.1038%2Fs41567-021-01232-0.

[12] James Nelson, Rajarshi Tiwari, and Stefano Sanvito. Machine learning density functional theory for the hubbard model. *Phys. Rev. B*, 99:075132, Feb 2019. doi: 10.1103/PhysRevB.99.075132. URL https://link.aps.org/doi/10.1103/PhysRevB.99.075132.

[13] Peter Schmitteckert. Learning DFT. *The European Physical Journal Special Topics*, 230(4):1021–1029, apr 2021. doi: 10.1140/epjs/s11734-021-00095-z. URL https://arxiv.org/pdf/2008.07923.pdf.

[14] Z E Musielak and B Quarles. The three-body problem. *Reports on Progress in Physics*, 77(6):065901, 2014. doi: 10.1088/0034-4885/77/6/065901. URL https://dx.doi.org/10.1088/0034-4885/77/6/065901.

[15] Essler Fabian H L. *The One-dimensional Hubbard model*, pages 1–10, 20–25. Cambridge University Press, 2005.

[16] Gerald D. Mahan. *Many-particle physics*, chapter 1.2. Springer US, 2000. ISBN 0306463385.

[17] T. Siro and A. Harju. Exact diagonalization of the hubbard model on graphics processing units. *Computer Physics Communications*, 183(9):1884–1889, 2012. ISSN 0010-4655. doi: https://doi.org/10.1016/j.cpc.2012.04.006. URL https://www.sciencedirect.com/science/article/pii/S0010465512001452.

[18] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45(4):255, 1950. doi: 10.6028/jres.045.026.

[19] K. Schönhammer, O. Gunnarsson, and R. M. Noack. Density-functional theory on a lattice: Comparison with exact numerical results for a model with strongly correlated electrons. *Phys. Rev. B*, 52:2504–2510, Jul 1995. doi: 10.1103/PhysRevB.52.2504. URL https://link.aps.org/doi/10.1103/PhysRevB.52.2504.

[20] U. Fano. Description of states in quantum mechanics by density matrix and operator techniques. *Reviews of Modern Physics*, 29(1):74–93, 1957. doi: 10.1103/revmodphys.29.74.

[21] Steven Weinberg. *Lectures on Quantum Mechanics*, chapter 3.6, page 77–81. Camabridge University Press, 2013.

[22] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943. doi: 10.1007/bf02478259.

[23] Chris M. Bishop. Neural networks and their applications. *Review of Scientific Instruments*, 65(6):1803–1832, 1994. doi: 10.1063/1.1144830.

[24] Dabal Pedamonti. Comparison of non-linear activation functions for deep neural networks on mnist classification task, 2018.

[25] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi: 10.1038/323533a0.

[26] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007. doi: 10.1007/s00365-006-0663-2.

[27] Noemi G. Gyori, Marco Palombo, Christopher A. Clark, Hui Zhang, and Daniel C. Alexander. Training data distribution significantly impacts the estimation of tissue microstructure with machine learning. *Magnetic Resonance in Medicine*, 87(2): 932–947, 2022. doi: https://doi.org/10.1002/mrm.29014. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.29014`.

[28] Cory Simon. Generating uniformly distributed numbers on a sphere, Feb 2015. URL `http://corysimon.github.io/articles/uniformdistn-on-sphere/`. (accessed 27/03/23).

[29] James Nelson, Luuk Coopmans, Graham Kells, and Stefano Sanvito. Data-driven time propagation of quantum systems with neural networks. *Phys. Rev. B*, 106: 045402, Jul 2022. doi: 10.1103/PhysRevB.106.045402. URL `https://link.aps.org/doi/10.1103/PhysRevB.106.045402`.

[30] Charu C. Aggarwal. *Neural networks and deep learning: A textbook*. Springer International Publishing, 2018.

[31] Francois Chollet et al. Keras, 2015. URL `https://github.com/fchollet/keras`.