Machine Learning for Infinite Neutron Matter at Finite Temperatures

Author: Alba Martínez Marimon
Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.

Advisors: Arnau Rios Huguet and Javier Rozalén Sarmiento

Abstract: In this paper, machine learning is used to model the imaginary part of the self-energy of a neutron-star binary system, which is approximated as infinite neutron matter. The performance of the model is evaluated by computing physical properties such as the momentum distribution and the total energy. The results obtained with the model show a high degree of accuracy when compared to theoretical values.

I. INTRODUCTION

With the recent detection of gravitational waves originated from the merging of a neutron-star (NS) binary system [1], the need for accurate modeling of neutron star matter has increased. NS binaries represent some of the most extreme environments in the universe, with temperatures of tens of MeV [2]. NSs are formed from the remnants of Type II supernovae, caused by the gravitational collapse of the core of a star with a mass M $\gtrsim$ 8M$_\odot$. The mass of NSs typically ranges from 0.1M$_\odot$ to about 3 M$_\odot$, with their radii being approximately 10–15 kilometers [3].

The study of NS binaries is key to a better understanding of nuclear matter, as some regions show similar physical conditions to those observed in atomic nuclei. This resemblance has led to a growing interest in infinite neutron matter modeling, with the proposal of new, realistic nucleon-nucleon potentials [4].

Given the complexity of modeling such systems, Artificial Neural Networks (ANNs) can be a powerful tool. ANNs are computational models inspired by the neural networks of the human brain, which use interconnected hidden nodes (neurons) to learn patterns within data. Specifically, in this project ANNs are used as a supervised machine learning technique, meaning they learn from labeled data and manage to predict unseen data. The interest in modelling with ANNs is that they require significantly less memory than the entire dataset, since only the parameters of the model need to be stored. Furthermore, they enable inference instead of a simple interpolation, due to their ability to learn patterns and handling high-dimensional data. Inference grants predictions on unseen data points outside the range of training data, which can lead to interesting extrapolations for $T = 0$ MeV when only finite temperature data is available. These kinds of generalizations are very useful for real-world applications, where future data may differ from past data [5].

This document is structured as follows. In section II, we provide the necessary physical background, followed by the theoretical computational framework in section III, where we briefly introduce ANNs. In section IV, we provide a more in-depth explanation of the model building process. Finally, section V compares the results of the physical values obtained with the original dataset and those obtained with the ANN.

II. THEORETICAL FRAMEWORK

Physical systems of neutron matter at non-zero temperatures, such as binary neutron star systems or neutron stars in the early stages of their evolution, exhibit complex many-body correlations that complicate their modeling. An approach to tackle this problem is the Self-Consistent Green's Function (SCGF) method [6].

The SCGF formalism is based on Green's functions or propagators, which are mathematical objects that represent the dynamics of particles and their interactions, and are often depicted using Feynman diagrams. We can characterize very different systems, like neutron stars and nuclei, with the same formalism [7]. Besides all these advantages, perhaps the most important in the context of astrophysics and infinite neutron matter is that SCGF methods can be formulated consistently at non-zero temperature [8].

Another key benefit of these propagators is that a number of observables can be easily derived from them. The one-body propagator in the energy domain can be expressed as [9]:

$$\mathcal{G}_k(z) = \int \frac{\mathrm{d}\omega}{2\pi} \frac{\mathcal{A}_k(\omega)}{z-\omega}. \tag{1}$$

In Eq. (1), z is a general complex energy variable. The spectral function $\mathcal{A}_k$ represents the probability of either adding or removing a particle with momentum k and energy ω to the infinite system [9]. This probability distribution is normalized to unity,

$$\int \frac{\mathrm{d}\omega}{2\pi} \mathcal{A}_k(\omega) = 1. \tag{2}$$

When this integral is adjusted according to the thermal population of the hole states, the resulting value represents the momentum distribution,

$$n_k = \int \frac{\mathrm{d}\omega}{2\pi} \mathcal{A}_k(\omega) f(\omega), \tag{3}$$

where $f(\omega)$ is the Fermi-Dirac distribution $f(\omega) = [1 + \exp[(\omega - \mu)/T]]^{-1}$, which explicitly depends on the chemical potential μ.

By normalizing the momentum distribution and considering the neutron's spin degeneracy $\nu = 2$, we can determine the system's density:

$$\rho = \nu \int \frac{\mathrm{d}^3 k}{(2\pi)^3} n_k. \tag{4}$$

The SCGF formalism provides us with access to the description of the macroscopic properties of the system. Within this method, the total energy per particle can be obtained by

$$\frac{\mathcal{E}}{A} = \frac{\nu}{\rho} \int \frac{\mathrm{d}^3 k}{(2\pi)^3} \int \frac{\mathrm{d}\omega}{2\pi} \frac{1}{2} \left\{ \frac{\hbar^2 k^2}{2m} + \omega \right\} \mathcal{A}_k(\omega) f(\omega) - \frac{1}{2} \langle \hat{W} \rangle, \tag{5}$$

where $\hbar$ is the reduced Planck constant, m is the mass and A represents the number of particles (in our case, m is the neutron mass and A is the number of nucleons).

The key concept in this paper is the self-energy Σ_k, which is directly related to the spectral function:

$$\mathcal{A}_k(\omega) = \frac{-2 \operatorname{Im} \Sigma_k(\omega)}{\left[\omega - \frac{k^2}{2m} - \operatorname{Re} \Sigma_k(\omega)\right]^2 + \left[\operatorname{Im} \Sigma_k(\omega)\right]^2}. \tag{6}$$

Σ_k is a complex function that accounts for the effects of the surrounding medium on a single particle in the system. Calculating Σ_k is challenging because it involves summing over an infinite number of complex diagrams that represent different interactions. In the SCFG method, the *ladder approximation* is used to include the subset that contributes the most to Σ_k [7]. The imaginary part is relatively easy to obtain because it involves integrals over energy and momentum that are easily evaluated [9]. In SCGF approaches, $\operatorname{Re} \Sigma_k(\omega)$ is often obtained from $\operatorname{Im} \Sigma_k(\omega)$ through the dispersion relation [9]:

$$\operatorname{Re} \Sigma_k(\omega) = \Sigma_k^{\infty} - \mathcal{P} \int \frac{\mathrm{d}\omega'}{\pi} \frac{\operatorname{Im} \Sigma_k(\omega')}{\omega - \omega'}, \tag{7}$$

where $\mathcal{P}$ denotes the principal value of the integral, and Σ_k^{∞} is the Hartree-Fock (HF) self-energy. This is an effective one-body interaction that considers the average of the two-nucleon $\langle \hat{V} \rangle$ (three-nucleon $\langle \hat{W} \rangle$) interactions over the Fermi sea of one (two) particle. It does not depend on energy, but it does in momentum, and it is defined as

$$\Sigma_k^{\infty} = \nu \int \frac{\mathrm{d}^3 k_1}{(2\pi)^3} \langle k k_1 | \hat{V} | k k_1 \rangle_A n_{k_1} \tag{8}$$
$$+ \frac{\nu}{2} \int \frac{\mathrm{d}^3 k_1}{(2\pi)^3} \int \frac{\mathrm{d}^3 k_2}{(2\pi)^3} \langle k k_1 k_2 | \hat{W} | k k_1 k_2 \rangle_A n_{k_1} n_{k_2}.$$

All these physical properties will be calculated here using data obtained both from SCGF simulations [8]. The difference with respect to the outputs of the ANN model quantify the accuracy of the latter.

III. ARTIFICIAL NEURAL NETWORKS

The computational tool used in this project is an ANN. An ANN is a model that approximates an undetermined function Φ, with $\Phi(\mathbf{x}_i) = \mathbf{y}_i$, given a dataset $D = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), ..., (\mathbf{x}_N, \mathbf{y}_N)\}$. Mathematically, Φ is a function that maps input vectors $\mathbf{x}_i \in \mathbb{R}^n$ to output vectors $\mathbf{y}_i \in \mathbb{R}^m$ through a series of affine transformations followed by non-linear activation functions [10]. The basic units are the hidden nodes (*neurons*), where these non-linear transformations are done. The simplest ANN, with one input and one output, can be expressed as:

$$\Phi(\mathbf{x}) = \sum_{i=1}^{N_{\text{hid}}} W_i^{(2)} \sigma \left(W_i^{(1)} \mathbf{x} + B_i^{(1)} \right) + B_i^{(2)}, \tag{9}$$

where N_{hid} is the number of hidden nodes. Matrices $W_i^{(1)}$ and $W_i^{(2)}$, and vectors $B_i^{(1)}$ and $B_i^{(2)}$, are the weights and biases associated to the connections of the node i in the input layer (1) and the output layer (2), respectively. These are the parameters we need to find and store as our model. The dimensions of the matrices are (N_{hid}^{j+1}, N_{hid}^{j}), which depend on the number of nodes of the actual layer j and the following layer $j+1$, and the vectors have N_{hid}^{j+1} length. σ corresponds to the non-linear, continuous and non-decreasing function applied to the output of the affine function $W_i^{(1)} x + B_i$.

To find the optimal parameters, we construct a loss function, which characterizes the deviation of our prediction from the target, and minimize it using gradient descent-based methods. These methods use the gradients of the loss function with respect to the parameters of the ANN (weights W_i and biases B_i). This is done in order to find the direction of steepest descent and how much each parameter is contributing to the error in predicting the output [10].

To calculate these gradients we use a backpropagation algorithm, which at its core, is the chain rule for partial differentiation. These gradients are then used to update the weights and biases, iteratively refining the network's predictions. [10]

The training process described above is performed on a subset of the dataset called the *training set*, which is used to learn the patterns in the data. The remaining data, where we only evaluate the model's performance, is referred to as the *validation set.* The reason of this separation of the dataset is that we need to ensure that the model generalizes well to unseen data, instead of just memorizing the training data.

IV. LEARNING IMΣ_k

The aim of this project is to build a model using an ANN trained on a dataset containing the imaginary part of the self-energy of an infinite neutronic system (ImΣ_k)

with fixed density $\rho = 0.16$ fm^{-3} at non-zero temperatures. The model will use three input vectors: temperature, momentum and energy.

Our dataset is obtained by using SCGF simulations with N3LO interaction [4], and consists of about 3 million data points. The importance of using an ANN lies in its ability to model the system with far less information, thereby reducing computational complexity.

The architecture of our best-performing model is represented in Fig. 1. This figure illustrates a 4-layer ANN with 22 hidden nodes in the first two layers and 20 hidden nodes in the last two layers. This model was chosen for its low loss and superior results calculating the physical properties discussed in section II compared to others with different configurations. An interesting observation during the training process was that deeper models (i.e., those with more layers) outperformed shallower models with one or two layers but a large number of nodes.

As for the activation function used in the hidden nodes (blue circles in Fig. 1), *Leaky ReLU* [11] was chosen for its usual good performance in ANNs. This non-linear function has small, positive gradient for negative inputs, and leaves positive inputs unchanged.

We used the *Smooth L1* loss function [12], which combines the benefits of L1 (absolute error) and L2 (squared error) losses, making it less sensitive to outliers and providing smoother optimization. As for the optimizer, the best one by far was *Rprop* [13], which implements resilient backpropagation algorithm. This stands out for its efficiency in training ANNs with large datasets, and its ability to dynamically adjust learning rates for individual parameters.

The dataset has been split up in the training set, with 80% of the data and a final loss of 0.004, and the validation set with the 20% left. There are no signs of overfitting (i.e., the ANN is not memorizing the data) because both subsets converge simultaneously to the same loss.

Regarding ReΣ_k, an approach similar to that employed in ImΣ_k was initially attempted. However, we did not obtain satisfactory results. Therefore, we have followed the conventional way of obtaining ReΣ_k in SCGF methods, using the dispersion relation, Eq. (7).

The computations are carried out using the Python library PyTorch, which integrates the implementation of all operations related to ANNs [14].

V. RESULTS

In the top panels of Fig. 2, the values of ImΣ_k obtained with the ANN are displayed. These are shown for three different temperatures to highlight the differences between the data and the ANN predictions, as well as for three fixed momenta. These values are near the Fermi momentum, which is of particular interest due to the significant physical phenomena that occur at this point [8]. As expected for ImΣ_k, it is negative across the entire energy domain and exhibits a peak at low momentum, with its maximum value approaching zero as the temperature decreases. ImΣ_k is related to the lifetime of particles within the system, so its size gives an idea of how rapidly particles transition into other states.

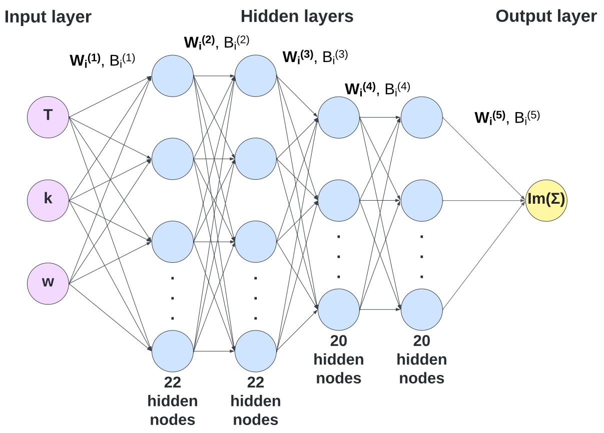

FIG. 1: Representation of the architecture of our ANN. The input layer, represented by purple circles, takes three input features: temperature (T), momentum (k), and energy (ω). In blue, we have *neurons*, where the non-linear transformations are done. Finally, the output layer is pictured in yellow.

These values of ImΣ_k have been used to obtain ReΣ_k through the dispersion relation in Eq (7). ReΣ_k is shown in the bottom panels of Fig. 2, also around the Fermi momentum and at different temperatures. ReΣ_k is related to the energy shift of a particle due to interactions within the system, so it is relevant to determine physical properties like the effective mass.

In Fig. 2 we notice that the ANN tends to work better at negative ω, specifically below ε_F (i.e., the energy at $T = 0$ MeV where ImΣ_k is zero). To quantify the differences between the discrepancies in the two regions of the energy ω domain, the relative errors in those are displayed in Table I.

TABLE I: Differences between the relative error in $\omega > \varepsilon_F$ and in $\omega < \varepsilon_F$ at high and low temperatures.

Energy ω (MeV)	**-200**		**200**	
T (MeV)	4	20	4	20
ImΣ_k relative error (%)	0.48	1.93	1.45	1.16
ReΣ_k relative error (%)	0.38	1.28	7.54	10.07

The lines in ReΣ_k are slightly oscillating at high energies, and this is probably related to the higher relative error in that region. This could be attributed to the computational expense of the integral we need to carry out, and a possibly insufficiently dense grid used for the calculation.

In order to give a physical meaning to the quality of the ANN, some microscopic quantities described in Section II have been computed. In this section, we compare these values obtained with SCGF methods and the ones

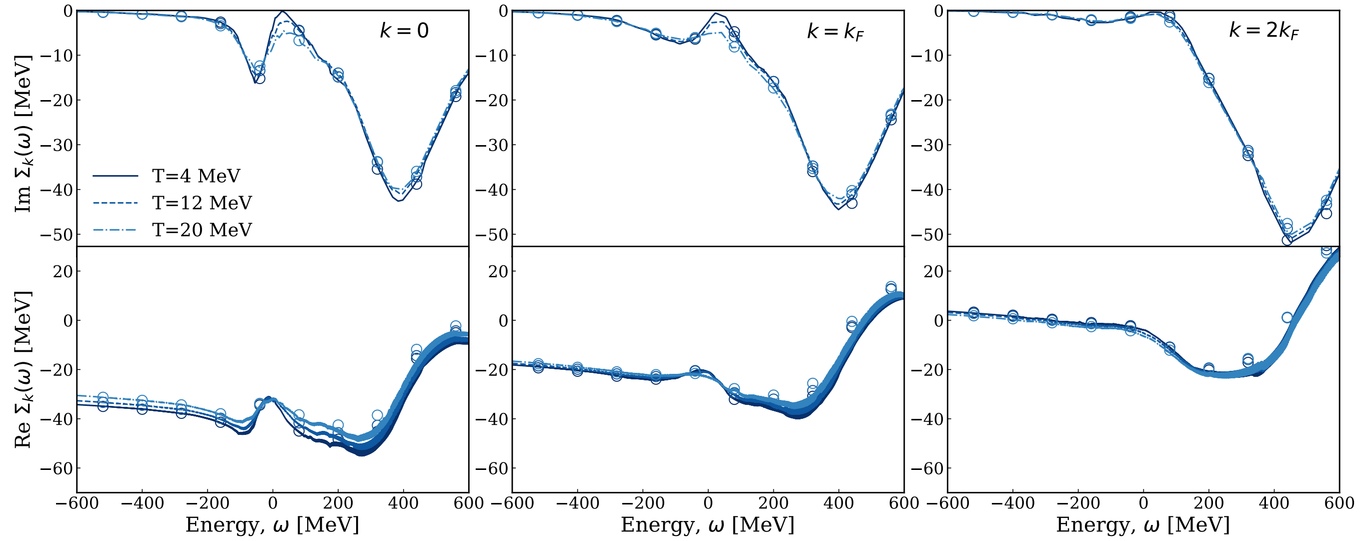

FIG. 2: Imaginary (top panels) and real (bottom panels) components of the self-energy of an infinite neutron-matter system at different fixed momenta $k = 0$ (left), $k = k_F = 331.41$ MeV (center) and $k = 2k_F = 662.82$ MeV (right). The symbols represent the data points obtained with the SCGF simulation, and the lines correspond to the results obtained by the ANN.

obtained with the predictions made by the ANN. The chemical potential μ and the three-body HF term Σ_k^∞, which are variables we need to calculate these quantities, are the same in both approaches and are the ones obtained with the SCGF simulation.

In Fig. (3) the occupation or momentum distribution near the Fermi momentum is displayed. As shown in Eq. (3), this is the result of normalizing the product of the spectral function and the Fermi-Dirac distribution, the latter being a step function at zero temperature. Therefore, n_k ranges from 0 to 1, representing the probability or fractional occupancy of particles in different states. Consistently with statistical mechanics, at lower temperatures, n_k is higher at low momenta, and as the temperature increases we have a broader distribution. This reflects an increment in the thermal energy, allowing particles to occupy higher momentum states.

The discrepancy between the ANN predictions (represented by lines) and the data points (represented by circles) is more pronounced at low momenta but shows an almost perfect match at high momenta. This might occur because the ANN struggles to capture the subtle interactions that occur at low momenta, or it may be due to an insufficient number of data points in this region.

The performance of the ANN in Fig. 3 is also slightly inferior at low temperatures. This is to be expected, given that n_k is calculated by integrating $\mathcal{A}_k$, which at low temperatures tends to a Dirac delta function. The integration of a delta function, or a very thin function, can present a significant computational challenge.

An effective way to evaluate the quality of the ANN is normalizing $\mathcal{A}_k$ and n_k, since the expected result of the integral is known (shown in Eqs. (2) and (4)). These normalizations are displayed in Table II. This demonstrates the accuracy of the results and how its quality slightly varies with temperature: in n_k, it diminishes at lower temperatures, while in $\mathcal{A}_k$ it improves. As mentioned early, the reduction of accuracy in the results for n_k at low temperatures could be due to numerical reasons, as evidenced by the less precise normalization at $T = 4$ MeV

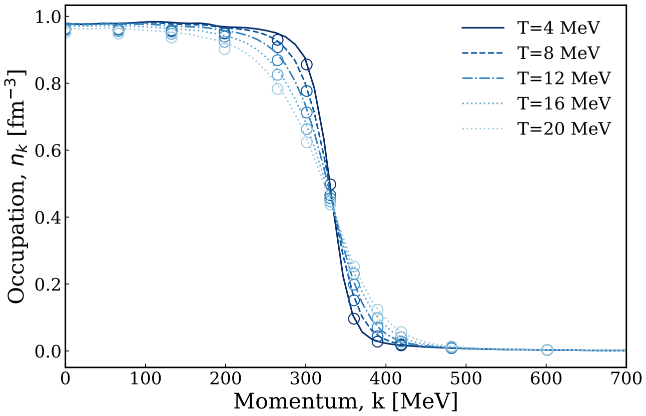

FIG. 3: Momentum distribution n_k at different temperatures. The symbols represent the data points obtained with the SCGF simulation, and the lines correspond to the results obtained with the ANN.

TABLE II: Average normalization values of occupation n_k and the spectral function $\mathcal{A}_k$ at different temperatures. The theoretical values are compared to the results obtained with the SCGF simulations and with the ANN.

Normalization	n_k			$\mathcal{A}_k$		
T (MeV)	4	12	20	4	12	20
Theoretical	0.1600	0.1600	0.1600	1.0000	1.0000	1.0000
SCGF	0.1611	0.1599	0.1599	1.0000	0.9999	0.9999
ANN	0.1635	0.1627	0.1618	0.9953	0.9942	0.9928

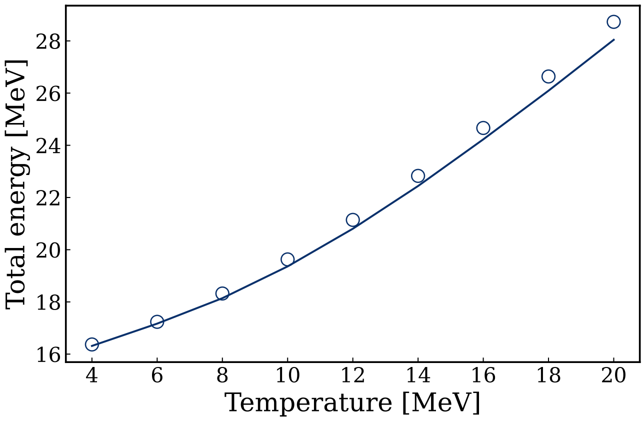

FIG. 4: Total energy per particle as a function of temperature. The symbols represent the data points obtained with the SCGF simulation, and the lines correspond to the results obtained with the ANN.

within the SCGF formalism.

The total energy per particle of the system at each temperature is also compared in Fig. 4. It has been computed using Eq. (5). As expected, the total energy shows a strong dependence on temperature, mainly due to the kinetic contribution. The results obtained with the ANN closely match the data points, indicated by blue circles. However, there is a decrease in accuracy as the temperature increases: the relative error of the model at $T = 4$ MeV is 0.34% and at $T = 20$ MeV it is 2.40%. This decline in accuracy may be due to the dependence of the energy per particle on $\mathcal{A}_k$, which aligns more closely with the data at lower temperatures.

VI. CONCLUSIONS

We have successfully reproduced the imaginary component of the self-energy for an infinite neutron system as a function of energy and momentum at non-zero temperatures using an ANN. We have then derived the real component of the self-energy and validated the model by computing some relevant physical properties. Key calculations of these properties made with the ANN, such as the normalization of the momentum distribution and the spectral function, show relative errors of 0.56% and 1.86%, respectively, compared to the dataset obtained from the SCGF simulation.

Our approach effectively reduced a dataset of 3 million points to a model with approximately 1.500 parameters, significantly decreasing the amount of storage needed. This reduction not only improves computational efficiency but also enables inference instead of interpolation. Another advantage of the ANN is that it learns patterns within the data and generally provides a better extrapolation compared to traditional methods. Future outlook would include exploring what the ANN predicts at $T = 0$ MeV.

Additional future work could involve further fine-tuning of the model. Adding more layers to the ANN could significantly improve its performance, as we observed it was a very important factor in the fine-tuning process.

This work opens the door to applying astrophysical simulations without relying on millions of data points. By using ANNs, we can achieve high-accuracy results at a much lower computational cost, which can lead to more efficient and accessible modeling of complex phenomena.

Acknowledgments

I would like to thank my advisor Dr. Arnau Rios for this year of immense learning and for his dedication to my work and guidance, as well as to Javier Rozalén for his willingness to help and his valuable advice. Additionally, I would like to thank Giuseppe Rivieccio for his insightful contributions to this project. Finally, I want to express my gratitude to my family and friends for their support throughout my degree, especially during the final stages.

[1] B.P. Abbott et al. *Phys. Rev. Lett.*, 119(16), 2017.

[2] Carolyn A. Raithel, Vasileios Paschalidis, and Feryal Özel. *Phys. Rev. D*, 104:063016, 2021.

[3] J. M. Lattimer and M. Prakash. *Science*, 304(5670):536, 2004.

[4] D. R. Entem and R. Machleidt. *Phys. Rev. C*, 68:041001, Oct 2003.

[5] N. Mojab, V. Noroozi, et al. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 677–684, 2020.

[6] A. Carbone, A. Rios, and A. Polls. *Phys. Rev. C*, 90:054322, 2014.

[7] C. Barbieri and A. Carbone. *An Advanced Course in Computational Nuclear Physics: Bridging the Scales from Quarks to Neutron Stars*, pages 571–644. Springer International Publishing, Cham, 2017.

[8] A. Rios. *Frontiers in Physics*, 8, 2020.

[9] A. Rios. PhD thesis, University of Barcelona, 2007.

[10] P. Mehta, M. Bukov, C.H. Wang, et al. *Phys. Rep.*, 810:1–124, 2019.

[11] J. Xu, Z. Li, et al. In *2020 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–7, 2020.

[12] M. Schmidt et al. In Joost N. Kok et al., editors, *Machine Learning: ECML 2007*, pages 286–297. Springer, 2007.

[13] M. Riedmiller and H. Braun. In *IEEE International Conference on Neural Networks*, pages 586–591 vol.1, 1993.

[14] A. Paszke, S. Gross, et al. *Proceedings of the 33rd International Conference on Neural Information Processing Systems.* Curran Associates Inc., 2019.