A Deep Dive into the Connections Between the Renormalization Group and Deep Learning in the Ising Model

Thesis by
Kelsie Taylor

In Partial Fulfillment of the Requirements for the
Degree of
Bachelor of Science in Physics

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2023
Defended June 5th

Kelsie Taylor
ORCID: 0009-0001-7510-2306

ACKNOWLEDGEMENTS

I would like to acknowledge my thesis advisors Dr. Joseph Lykken and Dr. Maria Spiropulu for making this project possible. I'd also like to acknowledge the assistance of Jonathan Booker for his programming of the Wolff algorithm in C and Damian Musk for his assistance in moving the Restricted Boltzmann Machines into TensorFlow. I'd also like to thank my academic advisor Dr. Mark Wise for being a useful resource for me all four years. Furthermore, I'd like to thank my former Caltech research mentors: Dr. Alan Weinstein, Dr. Derek Davis, Dr. Michele Papucci, Dr. Nikolai Lauk, and Dr. Neil Sinclair. Without them, I would not have developed the research skills necessary to complete this project. In addition, I'd like to thank the members of Dabney House and Venerable House for supporting me through the many long nights I had during both my thesis and my entire degree.

This thesis work was supported by the Department of Energy Office of High Energy Physics QuantISED program grant SC0019219 on QCCFP-QMLQCF; the Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics; the Caltech SURF program and the Brinson Foundation.

ABSTRACT

The renormalization group (RG) is an essential technique in statistical physics and quantum field theory, which considers scale-invariant properties of physical theories and how these theories' parameters change with scaling. Deep learning is a powerful computational technique that uses multi-layered neural networks to solve a myriad of complicated problems. Previous research suggests the possibility that unsupervised deep learning may be a form of RG flow, by being a layer-by-layer coarse graining of the original data. We examined this connection on a more rigorous basis for the simple example of Kadanoff block renormalization of the 2D nearest-neighbor Ising model, with our deep learning accomplished via Restricted Boltzmann Machines (RBMs). We developed extensive renormalization techniques for the 1D and 2D Ising model to provide a baseline for comparison. For the 1D Ising model, we successfully used Adam optimization on a correlation length loss function to learn the group flow, yielding results consistent with the analytical model for infinite N. For the 2D Ising model, we successfully generated Ising model samples using the Wolff algorithm, and performed the group flow using a quasi-deterministic method, validating these results by calculating the critical exponent ν. We then examined RBM learning of the Ising model layer by layer, finding a blocking structure in the learning that is qualitatively similar to RG. Lastly, we directly compared the weights of each layer from the learning to Ising spin renormalization, but found quantitative inconsistencies for the simple case of nearest-neighbor Ising models.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Number *Page*

LIST OF TABLES

Number *Page*

Chapter 1

INTRODUCTION

Within quantum field theory and statistical physics, it is often necessary to change parameters within a theory in order to treat infinities and correct for the effects of self-interactions. This technique is called renormalization. In general, renormalization arises when there are problems of scale, where parameters describing a process at short-distance scales may disagree with those describing a process at long-distance scales. This idea can be generalized, forming the concept of the renormalization group (RG): the apparatus which allows us to deal with multi-scale features in physics.

Most phenomena have features that are characterized by multiple scales, and near certain critical points, the dynamics of these phenomena become scale-invariant. This implies that we can take a theory with some parameters when examined at small scales near a critical point, and instead examine it at coarser scale with new, "renormalized" parameters. Usually, this results in fewer relevant parameters being needed to describe the dynamics. This process is called an "RG flow". RG flows show that even systems that are microscopically different can have similar macroscopic behaviors. Due to this, we can define a class of systems consisting of all systems with the same renormalization parameters: the "universality class" of a model. Models in the same class can apply to any number of fields, including statistical mechanics, quantum behavior, social dynamics, or even the stock market [15].

Deep learning is another useful technique used in all areas of physics. It uses multiple layers of representations to learn features directly from training data, allowing for massive improvements in image processing [11], language modeling [14], and more [16]. In particular, we focus on algorithms known as Deep Neural Networks (DNNs), graphical statistical models where each layer receives inputs from the layer before them. DNNs have shown massive success, but it is still not completely theoretically understood why they work so well.

In a 2014 paper [16], Mehta and Schwab argued that one reason DNNs work so well is because they perform an effective coarse-graining of the training data, in the same way that an RG flow does. In particular, they claimed that an exact mapping

exists between the RG flow of the one and two dimensional Ising models and a DNN comprised of stacks of "Restricted Boltzmann Machines" (RBMs) (one of the simplest possible DNNs). This claim was controversial [13, 19], warranting further discussion.

The Ising model was chosen because it is a simple model to which renormalization group flows can be applied. It is a classical model that is directly solvable in the one-dimensional case, with a simple Hamiltonian that only depends on spins, interaction strength, and magnetic field. Because the model is simple, we are able to more easily prove results for the Ising model, results which can be generalized to the entire Ising universality class. This includes models not just in physics, but also models in medicine, sociology, and economics [2, 4].

The one dimensional Ising Model RG flow can be solved exactly using easy formulas, although the model does not contain any non-trivial critical behavior. One can use this exact solution to show mathematically that RG flow has a one-to-one mapping with stacks of RBMs. One may also apply RBM learning to the 2D Ising model, yielding a weight plot that gives results similar to the two dimensional Ising RG flow [16] .

The existence of a connection between RG flow and deep learning could be a paradigm shift in not just the field of physics, but also in other data-driven fields that rely on similar renormalization or learning techniques. The work showing that there is a mapping between RBMs and RG flow in the one dimensional Ising model is promising, and the similarities in deep learning in the 2D Ising model to renormalization group flow is compelling. Other studies in group theory, renormalization, and deep learning also suggest a compelling connection [3, 17].

We thus re-examine the claims of [16] in detail, providing a "deep dive" into the qualitative and quantitative nature of this question. To do so, for the remainder of this chapter, we discuss in greater detail the specifics of RG flows and RBM learning to provide the necessary background for the rest of the work. Then, we develop techniques for renormalization group flows in the 1D and 2D Ising models to provide a baseline to which we compare our RBM models, with the 1D Ising model discussed in Chapter 2 and the 2D Ising model discussed in Chapter 3. Following this, we use RBM learning to reconstruct the Ising model, and perform a qualitative analysis of the weight structure in Chapter 4. Lastly, we combine the RG structure and RBM structures to perform a quantitative analysis in Chapter 5 and reach our general conclusions in Chapter 6. In addition, we include additional figures in

Appendices A and B.

1.1 Renormalization Group Techniques

For a general renormalization group flow, we have some set of couplings $\{K\}$ that fully characterize the system for one scale. Then, in applying the renormalization group transformation, we receive a new set of couplings $\{K'\}$ which preserve some properties of the system. In order to perform a renormalization group flow, we must be able to find all of the couplings $\{K'\}$ as functions of the old couplings $\{K\}$. These new couplings will define a new, renormalized model.

Within the universality class of the d-dimensional Ising model, we apply renormalization group flows via a block spin procedure: grouping spins together and assigning a new spin to them based on the old ones. These groups of spins each have a respective size b. Within the group flow, we expect the free energy of the system to follow the inhomogeneous transformation law

$$f(\{K\}) = g(\{K\}) + b^{-d} f(\{K'\}) \tag{1.1}$$

and the correlation length of the system to follow the transformation law

$$\xi(\{K'\}) = b^{-1}\xi(\{K\}), \tag{1.2}$$

either of which can be used to calculate the new coupling constants $\{K'\}$ [5].

We attempt to use these equations to perform accurate RG flows, in order to compare the group flow results we get to RBM learning results. We apply some basic learning techniques to perform accurate RG flows, first learning the 1D Ising model to check the method's validity, and then using it to calculate the RG flow of the 2D Ising model. We run the learning in TensorFlow, a Python package for machine learning [1], attempting to calculate the new coupling constants $\{K'\}$ in terms of the old ones, $\{K\}$. To do so, we provide a loss function using either the free energy or the correlation length, then provide the original free energy or correlation length of the system, along with the original coupling constants and guesses for the new coupling constants. Then, the machine learns how the coupling constant runs through Adam optimization, a standard machine learning optimization algorithm [10, 18].

We also can run our method in reverse: inputting into the machine the running of the coupling constant, and having it determine the Hamiltonian from this coupling constant. This will allow us to see if the basic learning algorithm can learn locality properties we consider essential to nature. We only did a simple exploration of this for the 1D Ising model, however.

1.2 The Restricted Boltzmann Machine

We use simple energy-based models called Restricted Boltzmann Machines (RBMs) to analyze the connection between the RG and deep learning. Our following discussion follows closely to that in [9, 16].

We consider RBMs that act on simple binary data drawn from a probability distribution $P(\{v_i\})$ with $\{v_i\}$ a set of N binary "visible" spins indexed by $i = 1, ..., N$. For example, the data could consists of a black and white picture, with black pixels being denoted as 1 and white pixels being denoted as 0. Similarly, the data could be Ising spins.

RBMs model the data by introducing a new set of "hidden" spins, $\{h_j\}$, a set of M binary spins indexed by $j = 1, ..., M$. These spins couple to the visible spins through the following energy function:

$$E(\{v_i\}, \{h_j\}) = \sum_j b_j h_j + \sum_{ij} v_i w_{ij} h_j + \sum_i v_i a_i, \tag{1.3}$$

where $\lambda = \{b_j, w_{ij}, a_i\}$ forms the variational parameters of the model, with w_{ij} falling between $-\infty$ and ∞. The joint distribution for a given set of $\{v_i\}$ and $\{h_j\}$ is then given in the expected form from statistical mechanics:

$$p_\lambda(\{v_i\}, \{h_j\}) = \frac{\exp(-E(\{v_i\}, \{h_j\}))}{Z}, \tag{1.4}$$

where Z denotes the relevant partition factor. We can thus define marginal distributions for both visible spins,

$$p_\lambda(\{v_i\}) = \mathrm{Tr}_{h_j} \left\{ \frac{\exp(-E(\{v_i\}, \{h_j\}))}{Z} \right\}, \tag{1.5}$$

and hidden spins

$$p_\lambda(\{h_j\}) = \mathrm{Tr}_{v_i} \left\{ \frac{\exp(-E(\{v_i\}, \{h_j\}))}{Z} \right\}. \tag{1.6}$$

From this, we can define new variational Hamiltonians as follows for the visible spins,

$$p_\lambda(\{v_i\}) = \frac{\exp(-H_{RBM}(\{v_i\}))}{Z}, \tag{1.7}$$

and the hidden spins

$$p_\lambda(\{v_i\}) = \frac{\exp(-H_{RBM}(\{h_j\}))}{Z}. \tag{1.8}$$

For unsupervised learning, RBM parameters are generally chosen to minimize the Kullback-Leibler divergence between the actual distribution $P(\{v_i\})$ and the estimated one $p_\lambda(\{v_i\})$:

$$D_{KL}(P(\{v_i\})||p_\lambda(\{v_i\})) = \mathrm{Tr}_{v_i}\left\{P(\{v_i\})\log\left(\frac{P(\{v_i\})}{p_\lambda(\{v_i\})}\right)\right\}. \tag{1.9}$$

Of course, when the RBM reproduces the visible data exactly,

$$D_{KL}(P(\{v_i\})||p_\lambda(\{v_i\})) = 0. \tag{1.10}$$

However, minimizing $D_{KL}(P(\{v_i\})||p_\lambda(\{v_i\}))$ cannot be done analytically in most cases. Instead, we minimize the Contrastive Divergence, the difference between two Kullback-Leibler functions, given by

$$CD_1 = D_{KL}(P(\{v_i\})||p_\lambda(\{v_i\})) - D_{KL}(P(\{v_i\})||p_{\lambda,1}(\{v_i\})), \quad [6, 8] \tag{1.11}$$

where $p_{\lambda,1}(\{v_i\})$ is a sample of the distribution $p_\lambda(\{v_i\})$. We minimize this equation because it is easier to approximate the gradient of this function than just the Kullback-Leibler function. Then, we can minimize the function using stochastic gradient descent with momentum, another standard machine learning optimization algorithm [18].

To do so, we convert our weight tensors to to binary spins. It suffices to generate 64 numbers n_{ij} between 0 and 1, then convert $n_{ij} > .5$ to 1 and the rest to 0. In particular, we convert the weight tensor values to the range $[0, 1]$ via the logistic function:

$$\sigma(w) = \frac{1}{1 + e^{-w}} \tag{1.12}$$

Thus, a large positive weight is close to 1, a large negative weight is close to 0, and a zero weight is .5.

We can thus construct a probability distribution from the hidden variables using

$$\mathrm{pHid}_j = \sigma(h_j), \tag{1.13}$$

and take a random sample from this distribution, denoted $\mathrm{sampHid}_j$. We can then define a new reconstructed lattice of

$$\mathrm{vReco}_i = \sum_j w_{ij}^T \cdot \mathrm{sampHid}_j \tag{1.14}$$

and reconstructed hidden weights of

$$\text{pHidReco}_j = \sum_i \sigma(w_{ij} \cdot \text{vReco}_i). \tag{1.15}$$

From these distributions, the approximate gradient used in learning is given by

$$\nabla w_{ij} \approx \langle v_i \otimes \text{pHid}_j - \text{vReco}_i \otimes \text{pHidReco}_j \rangle. \tag{1.16}$$

To make this learning "deep", these RBMs are stacked upon one another, such that the hidden layer of the first RBM is the visible layer of the second RBM, and so on. To do so, one maps a visible spin configuration to a hidden configuration using pHid_j. Then, the hidden spins' response to the visible spins can be treated as a new layer, and the cycle continues.

Chapter 2

PRELIMINARY CONNECTIONS IN THE ONE DIMENSIONAL ISING MODEL

2.1 Methods

In the one dimensional Ising model, the renormalization group flow is exactly solvable. In particular, this model contains N_0 classical spins, with each spin taking a value of either 1 or -1. The Hamiltonian is a nearest-neighbor Hamiltonian, given by

$$H_0 = -K \sum_{k=0}^{N_0-1} s_0^k s_0^{k+1}. \tag{2.1}$$

Here, we take $K = 1$ without loss of generality, absorbing it into the inverse temperature β_0. We assume periodic boundary conditions, in which $s_0^0 = s_0^N$. [5]

From the Hamiltonian, we may calculate the partition function

$$Z(\beta_0) = \mathrm{Tr}_{s_0} e^{-\beta_0 H_0}. \tag{2.2}$$

We may then calculate the free energy

$$f_0(\beta_0) = -\frac{1}{N_0} \log(Z(\beta_0)), \tag{2.3}$$

which has an analytical value of

$$f_0(\beta_0) = -\frac{1}{N_0} \log\left((2\cosh\beta_0)^{N_0} + (2\sinh\beta_0)^{N_0}\right). \tag{2.4}$$

We can also derive an analytical expression for the correlation length of the model, given by

$$\xi_0(\beta_0) = \frac{c}{\log\tanh\beta_0}, \tag{2.5}$$

where $c = -1$ for simplicity (this constant does not affect the group flow).

In applying the renormalization flow using blocks of b spins, we yield a new system with $N_1 = N_0/b$ sites and a new inverse temperature $\beta_1(\beta_0)$. Similarly to Equations 2.1-2.4, we may define for these new parameters a new Hamiltonian

$$H_1 = -K \sum_{k=0}^{N_1-1} s_1^k s_1^{k+1}, \tag{2.6}$$

a new free energy

$$f_1(\beta_1) = -\frac{1}{N_1}\log((2\cosh(\beta_1)^{N_1} + (2\sinh\beta_1)^{N_1}), \tag{2.7}$$

and a new correlation length

$$\xi_1(\beta_1) = \frac{c}{\log\tanh\beta_1}, \tag{2.8}$$

where $c = -1$ for consistency.

We thus have a way to calculate free energies and correlation lengths directly from the temperatures, allowing for them to be calculated in loss functions. To create a free energy loss function, we note that equation 1.1 becomes

$$f_0(\beta_0) = g_1(\beta_0) + f_1(\beta_1)/b, \tag{2.9}$$

and we can minimize the loss function

$$L(\beta_1) = f_0(\beta_0) - g_1(\beta_0) - f_1(\beta_1)/b. \tag{2.10}$$

Alternatively, we also know that

$$\xi_1(\beta_1) = b^{-1}\xi_0(\beta_0), \tag{2.11}$$

and we can thus minimize a loss function defined by

$$L(\beta_1) = b \cdot \frac{-1}{\log\tanh\beta_1} + \frac{1}{\log\tanh\beta_0}, \tag{2.12}$$

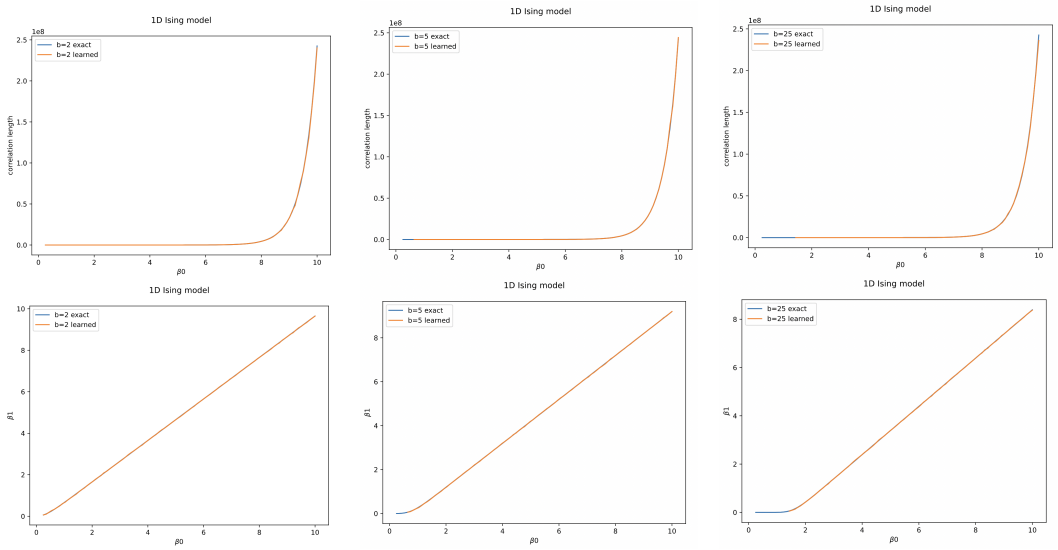
This running of the coupling constant is analytically solvable, so we can validate our results. We have that:

$$\beta_1(\beta_0) = \tanh^{-1}(\tanh(\beta_0)^b) \tag{2.13}$$

and

$$g_1(\beta_0) = \frac{\log(\cosh(\beta_1))}{b} - \log(\cosh(\beta_0)) - \frac{b-1}{b}\log(2) \quad [5]. \tag{2.14}$$

The main loss function we use for the 1D Ising model is the correlation length loss function, equation 2.12. We use equations 2.5 and 2.8 to attempt to learn equation 2.13 through this loss function. Alternatively, we may also use equation 2.10 as a free energy loss function. Then, we'd use equations 2.4, 2.7 to attempt to derive equations 2.13 and 2.14. In addition, we also use the analytical forms of equation 2.4, 2.7, 2.13 and 2.14, along with an L1 regularization loss function, to attempt to learn the Hamiltonian couplings in equation 2.6. Due to the fact that all of these cases have analytical solutions, we can then check our results against the analytical solutions to see how well or how poorly the learning has done.



(a) $b = 2$ (b) $b = 5$ (c) $b = 25$

Figure 2.1: Comparison of the analytical results and learned results from RG flow for different b values, using the correlation length loss function given in Equation 2.12. Graphs on the top indicate the difference between the real correlation lengths and the learned ones: a measure of how well the loss function is minimized. The bottom graph compares the actual running of the coupling constant β_1 to the learned value. We successfully learn the running of the coupling constant for each b value, while running into a few problems at low β_0.

2.2 Results

The results using the correlation length loss function given in Equation 2.12 are shown in Figure 2.1. For the most part, our learning model works perfectly, with no way to tell the difference between the learned function and the analytical function. The running of the coupling constant is successfully learned and this works for b values up to 100.

There are a couple of minor problems in this model, however. For whatever reason, the loss function has trouble with learning on the lower values of β, with the function returning undefined values instead of actual numbers. This is likely a computing issue. Additionally, the correlation length model only applies to infinite systems, as the analytical correlation length value is defined for a infinite system. Given that all our 2D Ising model results deal with finite systems, it would be useful to also have a model using a finite system.

We attempted to do this using the free energy loss function Equation 2.10, finding estimations for equations 2.13 and 2.14. We are able to do it for β_0 values between .2 and 2, for up to $N_0 = 1024$. Higher than that, we run into overflow problems in

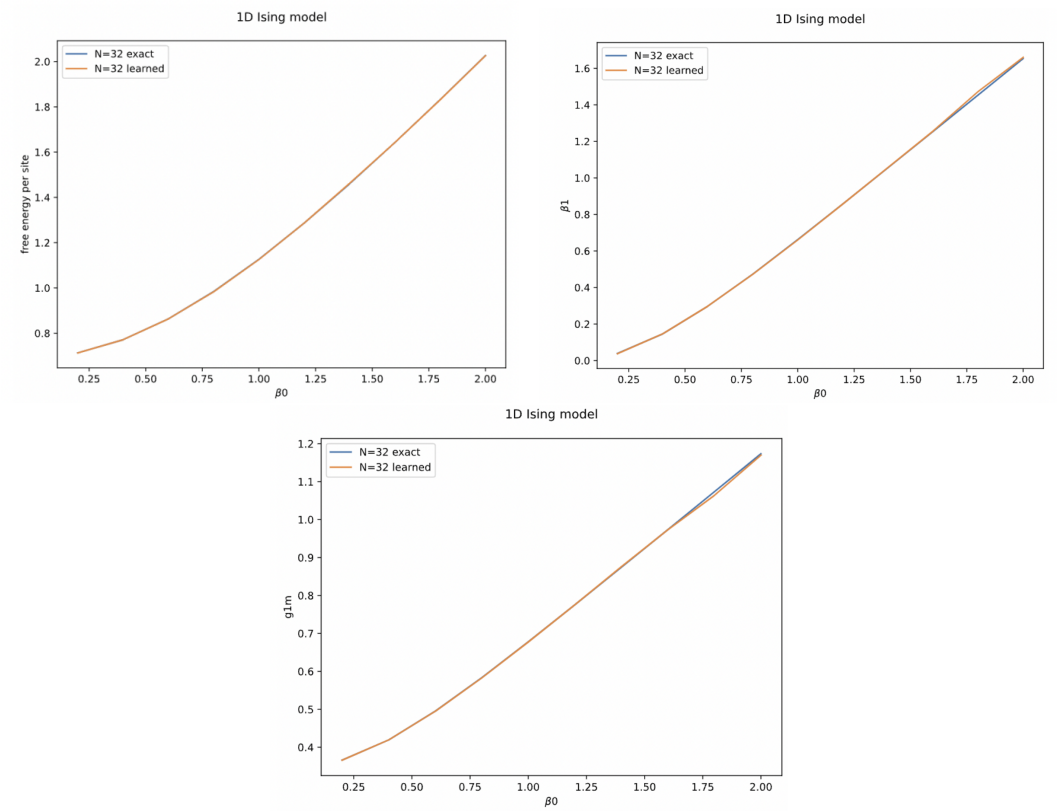


Figure 2.2: Comparison of the analytical results and learned results from RG flow for 1D Ising using the free energy loss function Equation 2.10 at $N_0 = 32$. We let $N_0 = 32, b = 2$, evaluated the loss function over four iterations of the renormalization group, guessed $\beta_1 = .2, g_1 = 0$, and ran over 1000 epochs at a learning rate of .005. The results for both β_1 and g_1 seem to match the analytical results very well, and so the learning is working successfully in this case.

the analytical solutions.

For the case of low N_0, we were able to yield learning results that were good. In Figure 2.2, we let $N_0 = 32$, $b = 2$, evaluated the loss function over four iterations of the renormalization group, guessed $\beta_1 = .2, g_1 = 0$, and ran over 1000 epochs at a learning rate of .005. As can be seen here, the results for both β_1 and g_1 seem to match the analytical results very well, and so the learning is working successfully in this case.

However, for most other cases, these results are less good, and are instead highly dependent on our starting values for β_1, g_1. In Figure 2.3, we let $N_0 = 512$, $b = 2$, evaluating the loss function over five iterations of the renormalization group, guessing $\beta_1 = .2, g_1 = 0$, and running over 1000 epochs at a learning rate of .005. These results are quite poor, with the model over-predicting β_1 and under-predicting

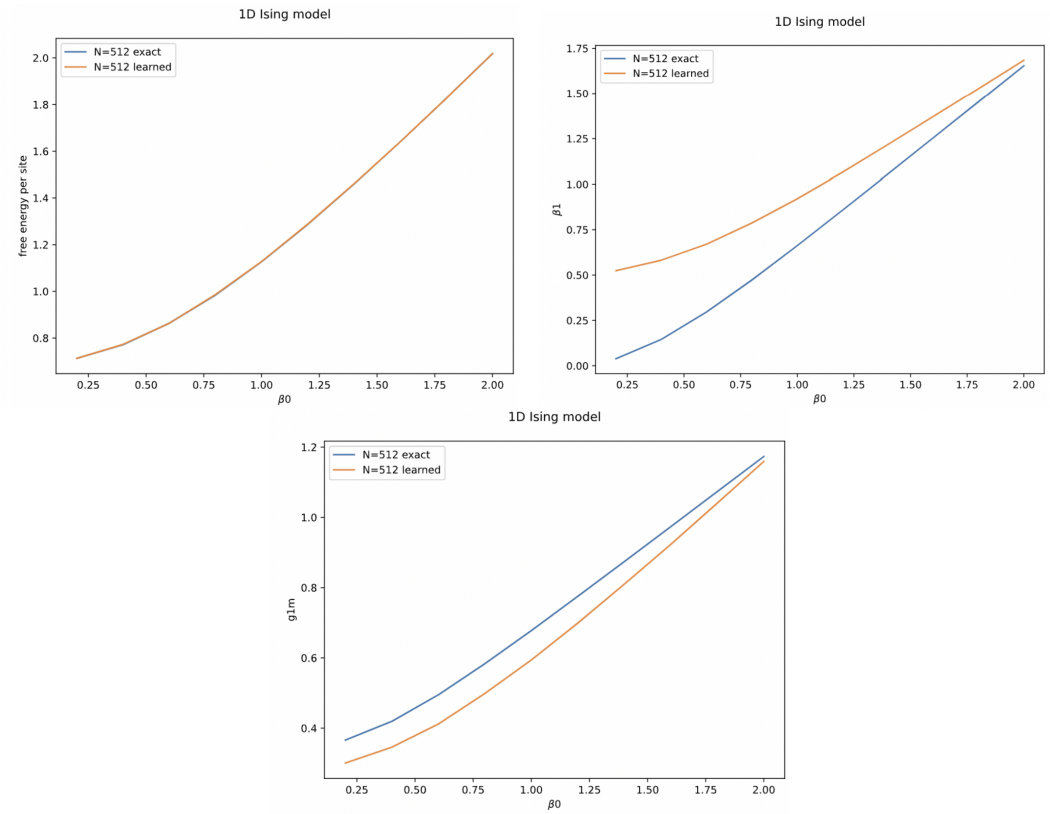


Figure 2.3: Comparison of the analytical results and learned results from RG flow for 1D Ising using the free energy loss function Equation 2.10 at $N_0 = 512$, a poor fit. The case where we let $N_0 = 512$, $b = 2$, evaluated the loss function over five iterations of the renormalization group, guessed $\beta_1 = .2, g_1 = 0$, and ran over 1000 epochs at a learning rate of .005. These results are quite poor, with the model over-predicting β_1 and under-predicting g_0, even though the free energy is minimized.

g_0, even though the free energy is minimized. Other results in other parameter spaces show similar patterns, with one of the parameters overestimated and the other underestimated. This usually means that very precise guesses are needed to properly learn the functions. For example, in the $N_0 = 512$ case, we can change our initial guesses to $\beta_1 = .833(\beta_0 - .2) + .1$ and $g_1 = .444(\beta_0 - .2) + .4$, yielding the results in 2.4.

These results suggest a general under-constraining of our learning that seems to be resultant from trying to calculate two values, β_1, g_1 from our singular loss function. This is, of course to be expected when using a single inhomogeneous equation to determine two functions. It is yet unclear as to how to resolve such an issue. Perhaps this method could be combined with using the correlation length method in order to learn both functions successfully.

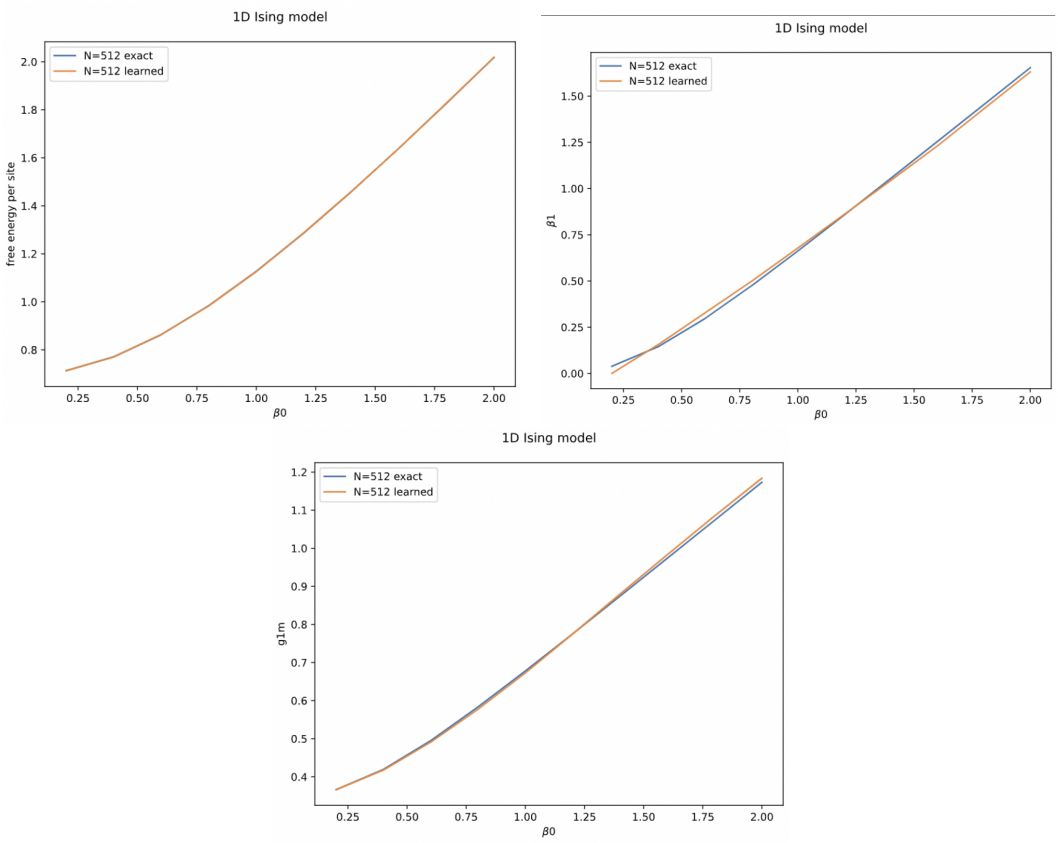


Figure 2.4: Comparison of the analytical results and learned results from RG flow for 1D Ising using the free energy loss function Equation 2.10 at $N_0 = 512$, a good fit. The case where we let $N_0 = 512$, $b = 2$, evaluated the loss function over five iterations of the renormalization group, guessed $\beta_1 = .833(\beta_0 - .2) + .1$, $g_1 = .444(\beta_0 - .2) + .4$, and ran over 1000 epochs at a learning rate of .005. These results are much better than those in Figure 2.3, showing the importance of good starting guesses to resolve under-constraining problems.

In addition, we also performed a method where we learned the renormalized Hamiltonian from the running of the coupling constant, letting the Hamiltonian be any possible coupling of the spin pairs and using the same free energy loss function as in equation 2.10. This was only run for the $N_0 = 16$ case, as higher spin values lead to computational problems that have not yet been resolved, due to the high amount of coupling constants. Here, we let β_0 go from .2 to 2, and initialize with a random coupling matrix with standard deviation of .2. We run for 2000 epochs with a learning rate of .001 and a L1 value of 5. The results of the learning are shown in

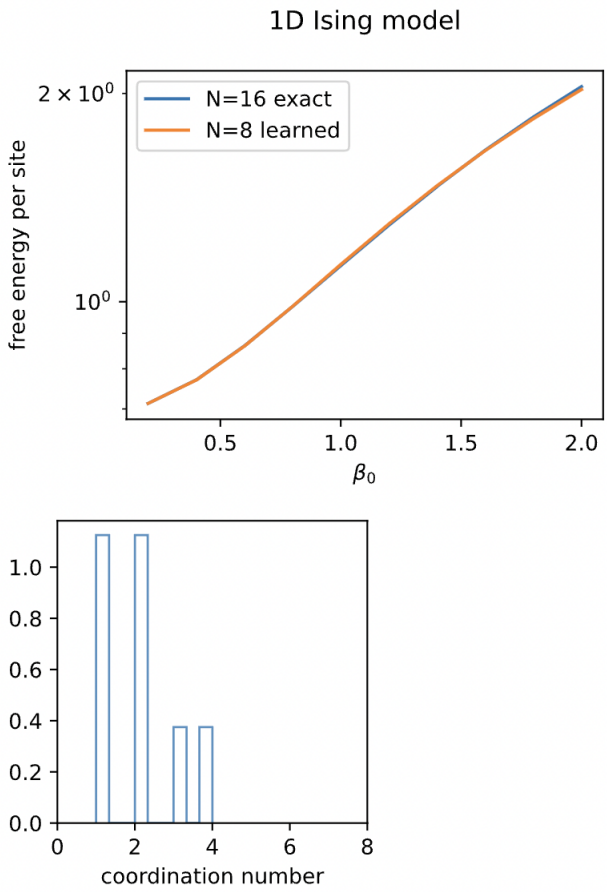


Figure 2.5: Learning the renormalized Hamiltonian. Here, we let β_0 go from .2 to 2, and initialize with a random coupling matrix with standard deviation of .2. Although the couplings aren't fully correct (they are given in equation 2.15), the resultant Hamiltonian does have an average non-zero correlation number of 2 and 8 total couplings, just like the 1D Ising model should.

Figure 2.5. The recovered coupling matrix is given by

$$\begin{pmatrix} 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1.07 \\ 0. & 0. & 0.84 & 0. & 0. & 0.84 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0.85 & 1.23 & 0. \\ 0. & 0. & 0. & 0. & 0.97 & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0.63 & 0.9 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{pmatrix} \tag{2.15}$$

Obviously, this result is not the same as the correct coupling matrix, but it does have an average non-zero correlation number of 2 and 8 total couplings, just like the 1D

Ising model. Thus, results seem promising to at least derive some dynamics from renormalization group flows, but more work should be done in this topic. Once again, perhaps this method could be combined with using the correlation length method in order to learn the coupling successfully.

Additionally, current progress in using the free energy loss function is hampered by overflow errors in the analytical solutions, making it impossible to reasonably calculate systems with $N_0 > 1024$ when we learn the group flow. Additionally, when we attempt to learn the Hamiltonian from the group flow, we run into larger scaling errors, as the number of parameters needed gets absurdly high for $N_0 > 16$. Further efforts may be done to expand the learning to as large of finite models as possible.

Chapter 3

RENORMALIZATION TECHNIQUES IN THE TWO DIMENSIONAL ISING MODEL

3.1 Methods

For the 2-dimensional Ising model, we consider a model with N^2 classical spins on an N by N lattice, where each spin $s[i, j]$ takes on a value $+1$ or -1. Using nearest neighbor couplings, we get a Hamiltonian of the form

$$H_0 = -K_1 \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \sum_{\langle nn \rangle} s[i, j]s[nn], \tag{3.1}$$

where $\langle nn \rangle$ means to sum over all nearest neighbors such that each possible spin-spin coupling only occurs in the sum once. The overall coupling constant K_1 we take to be 1, as we absorb it into the inverse temperature $\beta = 1/T$. We also assume periodic boundary conditions.

Unfortunately, for the two dimensional Ising model, there are not analytical solutions for the free energy, correlation length, or the general transformation laws in equation 1.1. This presents us with two main questions we must answer to perform the group flow: namely, how to create and validate accurate computational samples of the nearest-neighbor 2D Ising model and how to actually accomplish the renormalization. To generate samples, we use an algorithm known as the Wolff algorithm [12] and we use the critical exponent ν to validate it. We then perform the renormalization directly by changing the spins of each b by b block according to an algorithm, and then using either an analytical or machine learning method to yield $\beta_1(\beta_0)$. These two techniques are discussed below.

Wolff Algorithm

There are multiple Monte Carlo methods that are used in the generation of representative Ising model samples. Away from the critical temperature, the typical Metropolis algorithm works well for the Ising model. Here, a random spin is flipped, and then the algorithm either accepts or rejects the new spin with probability

$$p = \min(1, -e^{-\beta \Delta E}). \tag{3.2}$$

Over many epochs, the algorithm gives representative results at temperatures away from the critical temperature.

However, near the critical temperature, this algorithm is extremely slow and inefficient. Instead, we use a cluster algorithm called the Wolff algorithm, which is specifically designed for dealing with Ising model critical behaviors. This algorithm begins by choosing a random spin within the lattice, and then establishing bonds between nearest neighbors with the same spin with a probability of

$$p = 1 - e^{-2\beta J}. \tag{3.3}$$

It then proceeds to do this for each nearest neighbor spins, adding them to a stack and flipping their signs to avoid repeats. This proceeds until there are no more nearest neighbors or a stack maximum is hit. Over many epochs, this can quickly produce effective samples near the critical temperature for the 2D Ising model. [12]

In order to confirm that our models are representative of the 2D Ising model, we calculate the model's critical exponent ν from the data's correlation lengths. The correlation length is defined by computing the 2-point spin-spin correlator $\langle s(0)s(r)\rangle$, where $s(0)$ is defined as $s(0) = s[0,0]$ and $s(r)$ is a distance r away, where r is the shortest distance between the two spins. To find the correlation length, we then fit to the form

$$\langle s(0)s(r)\rangle = \left(\frac{a}{a+r}\right)^{1/4} e^{-r/\xi}, \tag{3.4}$$

where a is a parameter on the order of the lattice spacing, included to ensure $\langle s(0)s(r)\rangle = 1$. We choose $a = .2$ for a 64 by 64 grid, as we determined this value to fit the best empirically.

For the infinite nearest neighbor Ising model, we expect this correlation length to diverge at a critical temperature $T_c = 2.269$ or $\beta_c = .44$. As we approach this critical temperature, $\xi \to \infty$, and for $r >> a$ (as expected), we thus have

$$\lim_{T \to T_c} \langle s(0)s(r)\rangle = \frac{1}{r^{1/4}}, \tag{3.5}$$

which describes the universal behavior of the universality class of the 2D Ising model.

In order to then calculate the critical exponent ν, we can then fit the temperature dependence of the correlation length to the form

$$\xi(T) = c(T - T_c)^{-\nu}, \tag{3.6}$$

where simulated results show that $c \approx 1$. The critical exponent ν here should be 1 in the critical limit: where $N \to \infty$ and $T \to T_c$. [5]

By fitting to these critical exponent functions, we are able to see if the Wolff algorithm produces samples as expected; while noting that the effective critical temperature T_c may be slightly different due to the finiteness of our model.

Renormalization Group Flow

For the renormalization group flow, we take Ising models at temperatures slightly higher than T_c, such that we are still in the critical limit. We then proceed using block spinning of our N by N lattice, where we replace blocks of b by b spins with a single spin. The model thus becomes a lattice of N/b by N/b spins.

There are two methods we use for this block spinning algorithm. The first is a quasi-deterministic method, determined by the average of the spins in the b by b block. If the average is positive, we set the spin to 1. If it is negative, we set the spin to -1. If it is 0, we set it to 1 or -1 with equal probability.

The second method is called a "Hinton-like" method, as it is structured to be analogous to Hinton's RBM networks. In this method, for a given block, we define s_{av} as the average spin, and w as some nonnegative parameter. We then set the block spin to 1 with probability

$$P = \frac{1}{1 + e^{-s_{av} \cdot w}}, \tag{3.7}$$

and to -1 otherwise. If we let $w \to \infty$, this "Hinton-like" method then becomes the quasi-deterministic one. Due to this, we actually only implement the Hinton-method, but take w large to test out this quasi-deterministic method.

With this step complete, we must also calculate the running of the coupling constant β. However, this presents us with a problem. In the 2D Ising model, the nearest-neighbor Hamiltonian 3.1 does not, in general, block spin into a nearest-neighbor Hamiltonian. Instead, it block-spins into a model with next-to-nearest neighbor couplings. This new Hamiltonian is given by

$$H_0 = -K_1 \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \sum_{\langle nn \rangle} s[i,j]s[nn] - K_2 \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \sum_{\langle nnn \rangle} s[i,j]s[nnn], \tag{3.8}$$

where $\langle nnn \rangle$ means to sum over all next-to-nearest neighbors such that each possible spin-spin coupling only occurs in the sum once. This means we're actually in a 2D-space, with the running of $\beta \cdot K_1$ and $\beta \cdot K_2$. We choose to assume $\beta \cdot K_2 << \beta \cdot K_1$ and pay attention only to the running of $K_1 \cdot \beta$, but this does not include the full picture.

In order to calculate the running of the coupling constant, we consider two possible methods. The first method is using an analytical function developed by Maris and Kadanoff. For $b = 2$, we have that

$$\beta_1 = \frac{\alpha}{4} \log \cosh(4\beta_0), \text{ [7]} \tag{3.9}$$

where $\alpha = 1.604521$.

A second method to calculate β_1 is to use a method akin to that in Section 2.1, in which we minimize a loss function based on the correlation lengths. Using critical behavior described in equation 3.6, along with noting that equation 2.11 also holds in 2D, we can define a loss function given by

$$L(\beta_1) = b \cdot \frac{1}{1/\beta_1 - T_c} - \frac{1}{1/\beta_0 - T_c}, \tag{3.10}$$

and minimize it using Adam optimization to produce the running of the coupling constant.

From here, we can generate new Ising models that have gone through RG flow, and we can validate them through calculating the critical exponent ν as described in Section 3.1. We can use this to compare the different methods of block spinning and the different methods of calculating β, to find which is the most accurate.

3.2 Results

Wolff Algorithm

For the Wolff algorithm, we started by using Python code from Github that successfully generates 2D Ising model examples using the Wolff algorithm [20], later switching to C code that does the same thing but faster. One example of output provided by such code is given in Figure 3.1, which has $\beta = .395$ and is run over 10000 epochs. As seen in the figure, the Ising model keeps its macroscopic properties after 10000 epochs. Running the algorithm longer will not change the behavior, suggesting that 10000 epochs is enough to successfully train this value near the critical point. This is true for all examined β, except $\beta > .42$, which must run over 20000 epochs. Running over this many epochs is essential as it allows us to train many representative samples of the Ising model successfully.

For the validation of our model, we use a 64 by 64 Ising model. We first check that the results are consistent with the Ising model, calculating the critical exponent ν. The results are given in Figure 3.2. The results are good, yielding a ν value of 1.028

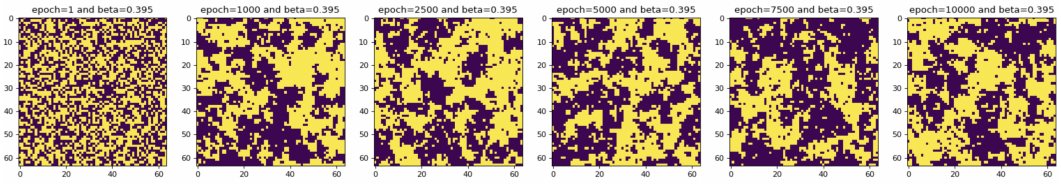


Figure 3.1: Generated example of the 2D Ising model from the Wolff algorithm for $\beta = .395$. The Ising model keeps its macroscopic properties after these 10000 epochs. Running the algorithm longer will not change the behavior, suggesting that 10000 epochs is enough to successfully train this value near the critical point. This is consistent for all β, except $\beta > .42$ must run for 20000 epochs.

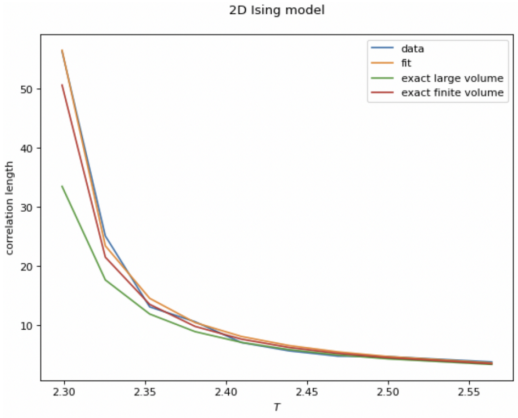
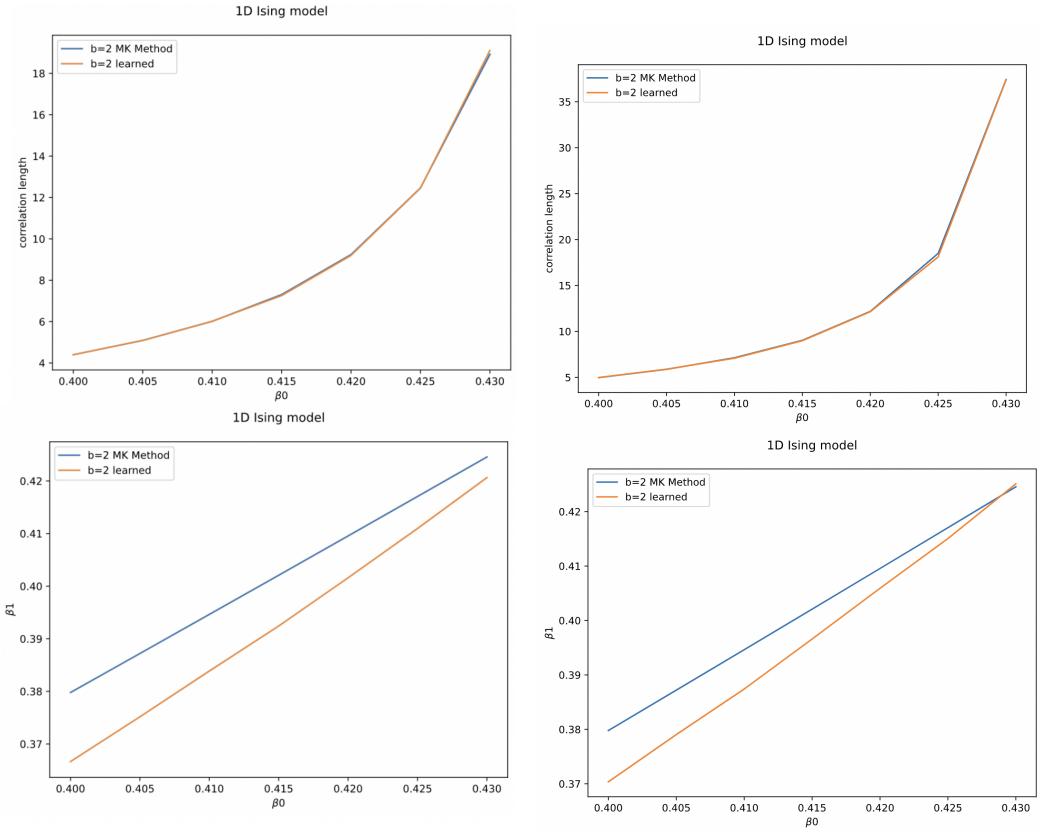


Figure 3.2: Fitting the temperature dependence of the correlation length, allowing for the calculation of ν. The fit here is good, yielding a ν value of 1.028 and a β_c value of .4388. The exact finite volume is calculated using the fitted value of β_c while the exact large volume uses the true value of $\beta_c = .44$. Results suggest a lower critical temperature due to finiteness, but seem to validate the model.

and a β_c value of .4388. This seems reasonably consistent with the true ν value of 1 and β_c value of .44.

However, models near criticality took much longer to run than expected for a critical temperature of .44. This suggests to us that the effective critical inverse temperature of the finite model is lower than .44, and is instead closer to .435. This is supported further by the fact that values of the correlation length in the data are larger than 32 for $\beta = .435$, and that our fit yields a value smaller than .44. This could suggest an effective critical temperature closer to .435, which we will find essential to note for Section 3.2.



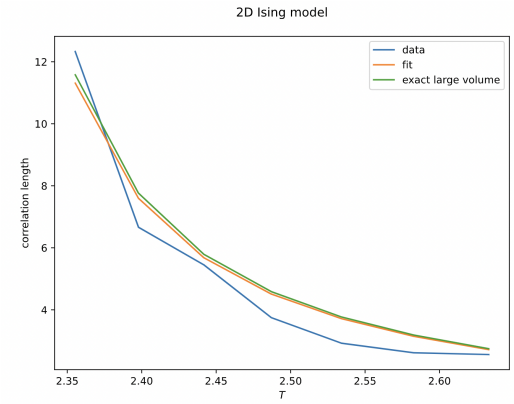
(a) Optimization Method with $\beta_c = .44$ (b) Optimization Method with $\beta_c = .435$

Figure 3.3: Results of using Adam optimization for β_1 compared to the Maris-Kadanoff method. The optimization seems to fit to the loss function well in both cases, as we would expect from Adam optimization. However, it is closer to Maris-Kadanoff when we take a lower inverse critical temperature of $\beta_c = .435$ than $\beta_c = .44$, yet is still very different.

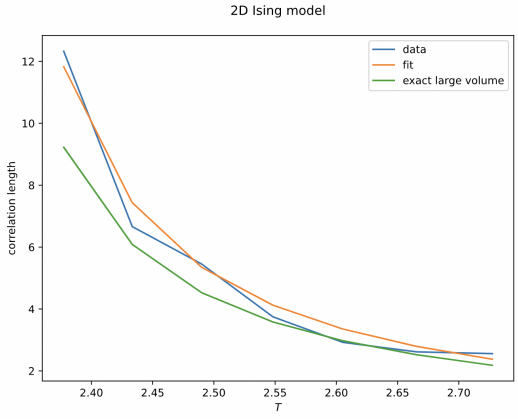
Renormalization Group Flow

We proceeded using the validated model from Section 3.2 and performed four steps (64 to 4) of a $b = 2$ RG flow on it for values of $w = 20$, $w = 5$ and $w = 1$. There are several conclusions we can draw from the results of our flow, which are summarized below.

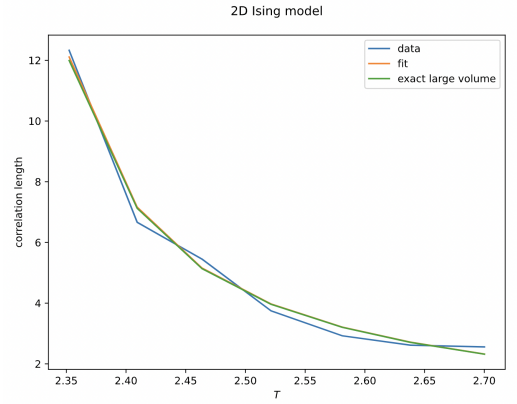
The first conclusion we reach is that the best method for calculating the running of the coupling constant β is given via optimization using the effective critical temperature *of the finite case*. When we went through the first step of the group flow, we compared fit results using three different values of β_1: those calculated via the Maris-Kadanoff equation, Equation 3.9, those using Adam optimization assuming $\beta_c = .44$, and those using Adam optimization assuming $\beta_c = .435$ (as suggested by results from Section 3.2). Comparisons of the β_1 values are shown in Figure 3.3 and



(a) Maris-Kadanoff Method



(b) Optimization Method with $\beta_c = .44$



(c) Optimization Method with $\beta_c = .435$

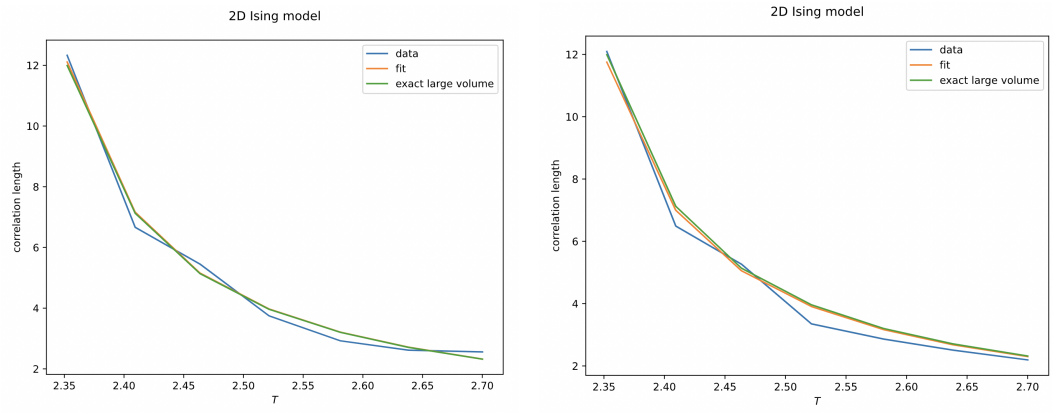
Figure 3.4: Fitting the temperature dependence of the correlation length of renormalized data, allowing for the calculation of ν, by method of calculating β_1 values. All figures are calculated for $N = 64$ and $w = 20$. The ν value for Figure 3.4a is $\nu = .986$, for Figure 3.4b is $\nu = 1.107$, and for Figure 3.4c is $\nu = 0.999$. The fits seem to be best when we use learning methods with a lower critical inverse temperature of $\beta_c = .435$, and worst when we use it with $\beta_c = .44$. Maris-Kadanoff is in the middle of the two, suggesting the importance of accounting for finiteness lowering the inverse effective critical temperature.

the actual fits of these values are shown in Figure 3.4, where we assume $w = 20$. The ν value for Maris-Kadanoff is $\nu = .986$, for $\beta_c = .44$ optimization is $\nu = 1.107$, and for $\beta_c = .435$ optimization is $\nu = 0.999$. The fits seem to be best when we use learning methods with a lower critical inverse temperature of $\beta_c = .435$, and worst when we use it with $\beta_c = .44$. Maris-Kadanoff is in the middle of the two. This suggests that the finite nature of our Ising model pushes the effective critical temperature closer to $\beta_c = .435$ than $\beta_c = .44$

The second conclusion we reach is that block spinning in the pseudo-deterministic limit is more accurate than Hinton-like block spinning. When we went through the first step of the group flow, we compared fit results using three different values of w: $w = 20$, $w = 5$, $w = 1$. These results for using $\beta_c = .435$ optimization are shown in Figure 3.5. The ν value for Figure 3.5a is $\nu = .999$, for Figure 3.5b is $\nu = .987$, and Figure 3.5c is $\nu = 0$. The fits seem to be best when we use methods in the pseudo-deterministic limit, and that they get worse as we get more and more Hinton-like. This can lead to dramatic problems in the fit for low w; $w = 1$ is not a valid renormalization group flow for the Ising model. This suggests problems with RBM learning, if it is more similar to low w Hinton learning than high w Hinton learning.

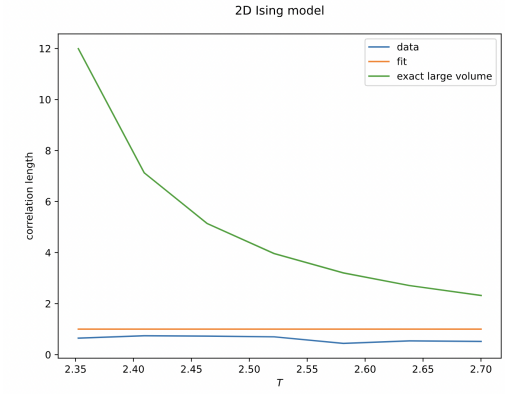
Lastly, we can conclude that the renormalization group flow is pretty accurate for the first two steps, assuming $w = 20$ and we use $\beta_c = .435$ optimization to calculate the β_1 values. After that, the renormalization group flow seems to get less reasonable. This can be seen in Figure 3.6 both visually and numerically, as the ν value for Figure 3.6a is $\nu = .999$, for Figure 3.6b is $\nu = 0.93$, for Figure 3.6c is $\nu = .857$, and for Figure 3.6d is $\nu = .698$. As the renormalization group flow continues, the fits get worse and worse, with a reasonable renormalization group flow for 2 iterations, but not for the last two.

This of course, is expected, as we are working in a one-dimensional parameter space when we should be working with a higher dimensional one. Allowing for next-to-nearest neighbors, or changing critical temperatures in the optimization for each layer may fix this.



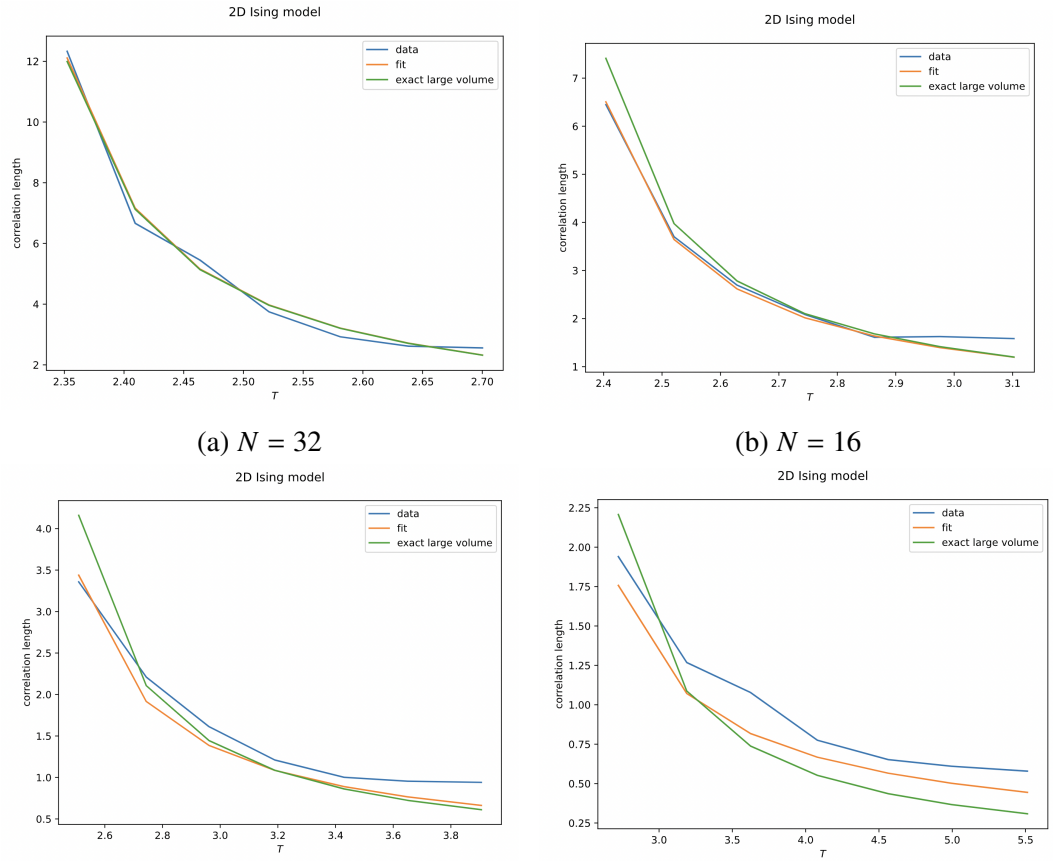
(a) $w = 20$

(b) $w = 5$



(c) $w = 1$

Figure 3.5: Fitting the temperature dependence of the correlation length of renormalized data, allowing for the calculation of ν, by value of w. All figures are calculated for $N = 64$ and using optimization with $\beta_c = .435$. The ν value for Figure 3.5a is $\nu = .999$, for Figure 3.5b is $\nu = .987$, and Figure 3.5c is $\nu = 0$. The fits seem to be best when we use methods in the pseudo-deterministic limit, and that they get worse as we get more and more Hinton-like. This can lead to dramatic problems in the fit for low w; $w = 1$ is not a valid renormalization group flow for the Ising model.



(a) $N = 32$ (b) $N = 16$

(c) $N = 8$ (d) $N = 4$

Figure 3.6: Fitting the temperature dependence of the correlation length of renormalized data after multiple renormalization group flow steps, allowing for the calculation of ν, by value of N. All figures are calculated for $w = 20$ and using optimization with $\beta_c = .435$. The ν value for Figure 3.6a is $\nu = .999$, for Figure 3.6b is $\nu = 0.93$, for Figure 3.6c is $\nu = .857$, and for Figure 3.6d is $\nu = .698$. As the renormalization group flow continues, the fits get worse and worse, but this is expected as we are restricting the RG flow to one dimension. We have a reasonable RG flow for 2 iterations, with it getting a bit worse after that. Allowing for next-to-nearest neighbors, or double-checking critical temperatures may fix this.

Chapter 4

ANALYSIS OF THE DEEP LEARNING OF THE TWO DIMENSIONAL ISING MODEL

4.1 Introduction

We now leave our discussion of the renormalization group to discuss the coarse-graining of the Ising model through RBM learning. To do so, we implemented a three-layer stack of Restricted Boltzmann Machines as described in Section 1.2, the same way as in [16]. For simplicity and speed, we made these RBMs in TensorFlow.

We made 20,000 instantiations of a 64x64 = 4096 spin Ising model for varying β values, produced by the Wolff algorithm discussed in 3.1. The first layer of the RBM had 32x32 = 1024 nodes, the second had 16x16=256 nodes, and the third had 8x8=64 nodes. In this way, we mirrored the renormalization procedure as discussed in Chapter 3.

The parameters of the learning are given in Table 4.1, chosen to match standard deep learning procedures and the procedures in [16] to effectively learn the coarse-graining of the Ising model. In particular, we use a non-zero L1 penalty on the weights to sparsify the results and penalize over-fitting. Some authors consider this to bias the results towards learning locality [13], but our goal is to more rigorously

Parameter	Value(s)
β Values	{.395, .4, 45, .41, .415, .42, .425, .43}
Number of Lattices	20000
Epochs in Wolff Algorithm	10000 if $\beta \leq .42$, 20000 if $\beta \geq .425$
Epochs in Learning	1500
Learning Rate	.004
Learning Rate Reduction	.998
L1 Regularization Weight	.0008
Batch Size	200
Momentum	.9

Table 4.1: Deep learning parameters for the RBM stacks. Parameters were chosen to best reproduce the learning in [16]. In particular, we apply an L1 regulator to penalize over-fitting, as is done in [16].

examine the claims made in [16] with the regularization, so we continue to use the regulator.

For the remainder of this chapter, we will discuss the results of this coarse-graining on its own, in detail. Then, in Chapter 5, we discuss the results via numerical comparisons to the previously discussed renormalization results.

In this chapter, we first discuss the reproduction of plots from [16] in detail in 4.2. We then discuss the structure of the trained weights and how they can form blocks in 4.3, followed by a deeper analysis of distances between spins in these blocks in 4.4. We conclude by using these results to analyze the use of the RBM network as an autoencoder in 4.5. In the following analysis, we only present plots from our $\beta = .41$ results, as this is the closest to the $\beta = .408$ used in [16]. However, we will discuss how these results vary across β values. We also present the full set of plots in Appendix A.

4.2 Receptive Field Analysis

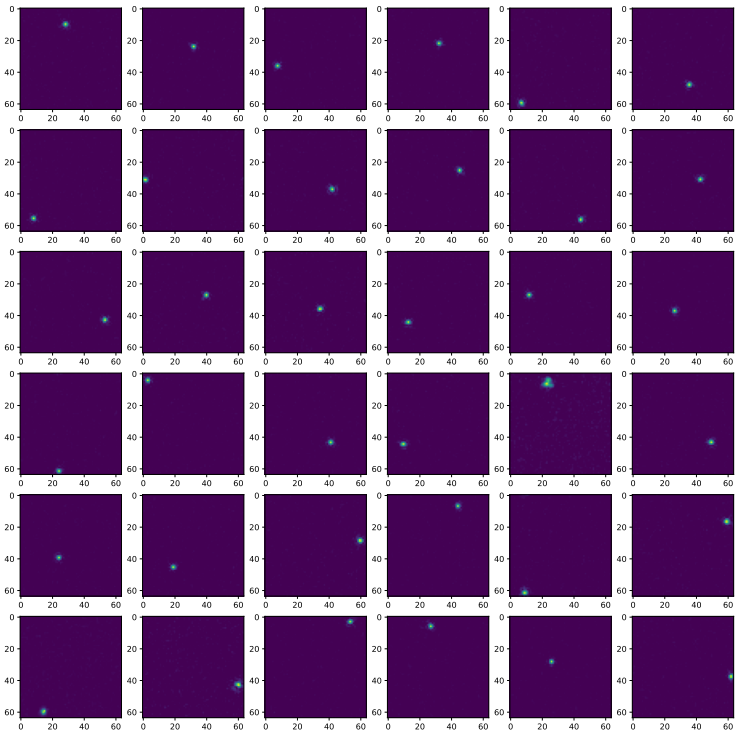


Figure 4.1: Layer 1 receptive field plot for $\beta = .41$. Each individual plot represents a node in Layer 1 and the values on the plot represents the full set of absolute weights corresponding to that node. The clustering of non-zero weights next to each other suggests the model is qualitatively learning some form of locality.

One of the main qualitative results from [16] were the "receptive field plots", which

we have reproduced in Figures 4.1 and 4.2 for $\beta = .41$. To understand what these plots represent, consider Layer 1. Each of the 32x32 = 1024 hidden nodes has an associated tensor of 64x64= 4096 weights, one for each Ising spin input, with each weight falling between $-\infty$ and ∞. Figure 4.1 is a color map of the absolute values of these weights for the first 36 nodes. Most of the 4096 weights are close to zero, with a few O(1) weights that are clustered together in terms of the original lattice.

To yield a similar map for other layers, we define a tensor similar to that from Layer 1: one which contains information on which of the original Ising spins fed information to each Layer 3 hidden spin. To do so, we denote the Layer 3 weights by $w^{L3}_{i_3 j_3; i_2 j_2}$ where indices i_3, j_3 run 1, ...8 and i_2, j_2 run 1, ...16. In a similar vein, we denote the Layer 2 weights by $w^{L2}_{i_2 j_2; i_1 j_1}$ and the Layer 1 weights by $w^{L2}_{i_1 j_1; ij}$, where i_1, j_1 run 1, ...32 and i, j run 1, ...64. We can thus define the Layer 3 "receptive field tensor" as

$$\mathrm{rec}^{L3}_{i_3 j_3; ij} = \sum_{i_2, j_2, i_1, j_1} w^{L3}_{i_3 j_3; i_2 j_2} \cdot w^{L2}_{i_2 j_2; i_1 j_1} \cdot w^{L1}_{i_1 j_1; ij}. \tag{4.1}$$

Similarly, we can define the Layer 2 and Layer 1 receptive field tensors as

$$\mathrm{rec}^{L2}_{i_2 j_2; ij} = \sum_{i_1, j_1} w^{L2}_{i_2 j_2; i_1 j_1} \cdot w^{L1}_{i_1 j_1; ij}, \tag{4.2}$$

and

$$\mathrm{rec}^{L1}_{i_1 j_1; ij} = w^{L1}_{i_1 j_1; ij}. \tag{4.3}$$

Thus, we can define receptive field maps for every layer. In particular, we examine the Layer 3 receptive field, which is given for all Layer 3 spins, in Figure 4.2. The Layer 3 receptive field plot is similar in structure to the Layer 1 receptive field plot, except with larger weight clusters. This structure is similarly found for receptive field plots of all the tested β values, as shown in Appendix A.1.

These figures form the basis of the argument in [16] that the trained network is accomplishing a variation of Kadanoff block spinning layer by layer. In this interpretation, each cluster of spins in the receptive field maps corresponds to a block in the Ising lattice, and each RBM node can be converted into a form equivalent to an Ising block-spinning.

We further clarify this argument by considering what an analogous plot would look like using RG techniques instead of the RBM. To do so, we take our renormalization

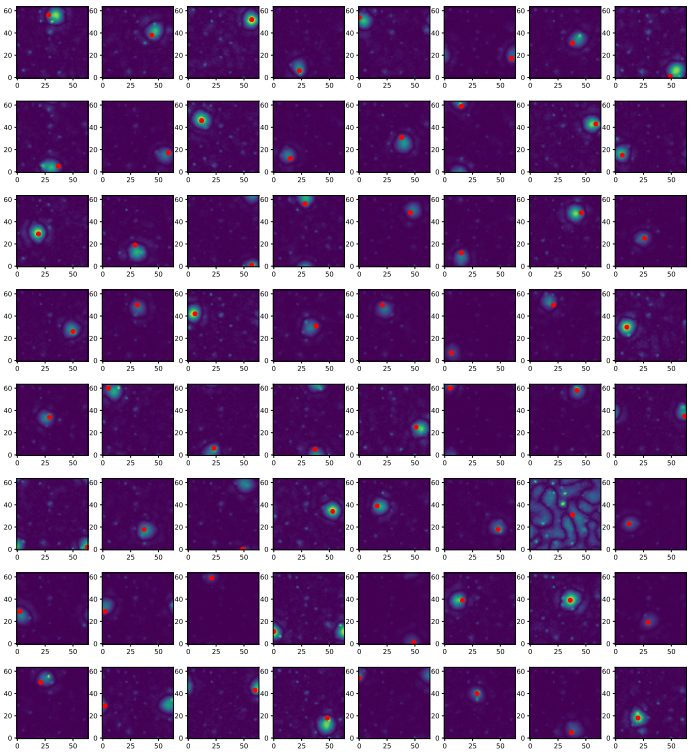


Figure 4.2: Layer 3 receptive field plot for $\beta = .41$. Each individual plot represents a node in Layer 3 and the values on the plot are those of a tensor representing the original Ising spins that fed information to the node. The clustering of non-zero values next to each other suggests the model is qualitatively learning some form of locality.

procedure from Chapter 3 and block spin the results from an initial 64 by 64 lattice to a 4 by 4 lattice. We then determine how the initial 64 by 64 spins affect a given spin in the final 4 by 4 model as follows: if an initial spin is both part of the block that determines the final spin and the same as the final spin, it gets a value of 1, otherwise it gets a value of 0; then, we average over all samples. These results for $\beta = .41$ averaged over 1000 samples with $w = 20$ and using optimization techniques are shown in Figure 4.3.

In this plot, we see a situation similar to that in Figures 4.1 and 4.2. Each node is only determined by a small block of the original 64 by 64 lattice, as expected from block spinning. In this way, we find that qualitatively, RBM learning looks like block spinning. Additionally, this qualitative connection holds for all the β values we tested, outside of just the $\beta = .408$ used in [16].

In some ways, this result is already surprising, given that the RBM network only uses 1D vector representations of the tensors, but seems to derive locality in the

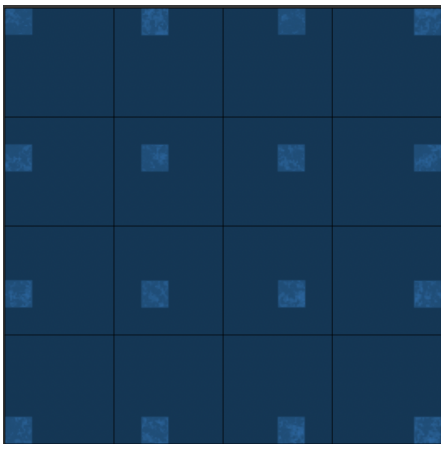


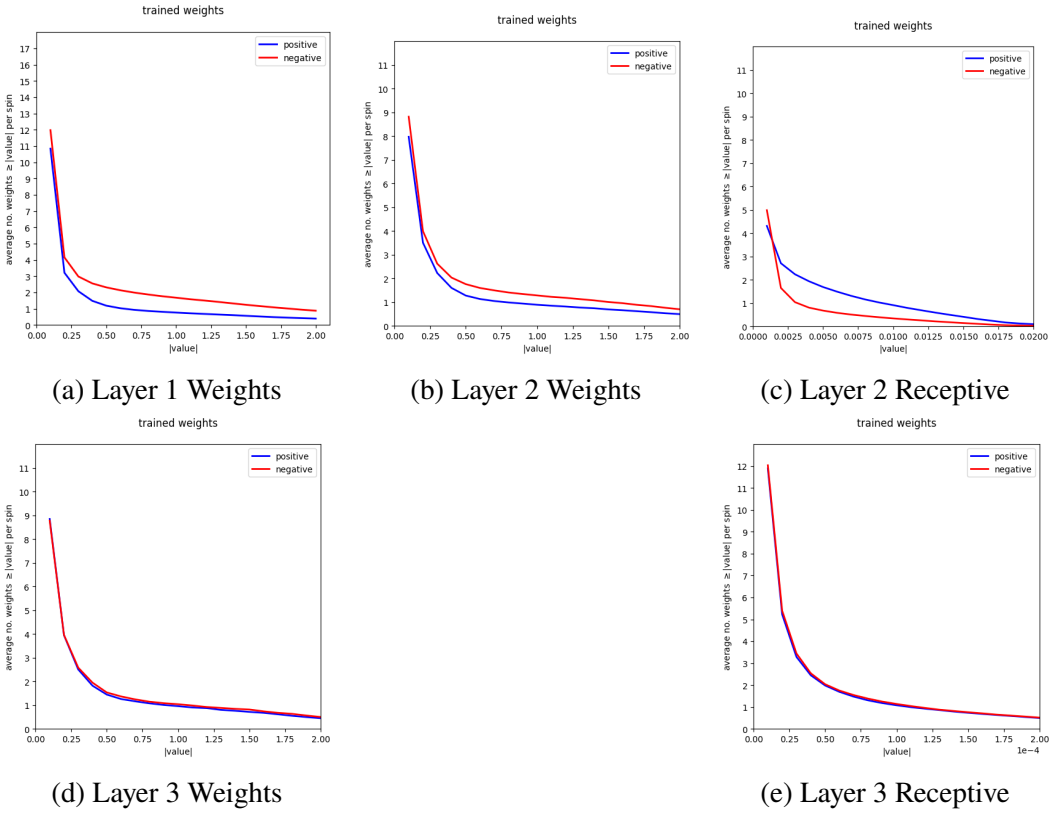
Figure 4.3: Plot for RG analogous to an RBM receptive field plot. We let $\beta = .41$ averaged over 1000 samples with $w = 20$ and using optimization techniques. We determined how the initial 64 by 64 spins affected a given spin in the final 4 by 4 model as follows: if an initial spin is both part of the block that determines the final spin and the same as the final spin, it gets a value of 1, otherwise it gets a value of 0; then, we average over all samples.

receptive fields. We further examine this locality qualitatively in the next sections.

4.3 Structure of Weights

To better understand the structure of this derived locality, we look at the weight tensor structure directly. We first note that the bias weights a_i, b_j, as described in Section 1.2, are 0 in our learning. This means that we only need to examine the properties of the weight tensors w for each layer. For layers 2 and 3, we consider the properties of both the weight tensor and the receptive field tensor separately, as we should consider the weight properties both alone and in the context of the layers above them.

For layer 1, we take each of the 64x64 = 4096 spin locations, examine the 32x32=1024 layer 1 weights connected to it, and calculate how many of these nodes have a positive spin or a negative spin greater than a given magnitude. We do the same thing for the layer 2 and 3 receptive field tensors so that for each of the 64x64 = 4096 weights, we examine the 16x16=256 layer 2 receptive field tensors and 8x8=64 layer 3 receptive field tensors. In addition, we can also consider the layer 2 and 3 weights instead of the receptive field tensors, so that for each of the 32x32=1024 spins fed into layer 2, we examine the 16x16=256 layer 2 weights and for each of the 16x16=256 spins fed into layer 3, we examine the 8x8=64 layer 3 weights. In each case, we then average over all the lattices to reach the plots in Figure 4.4.



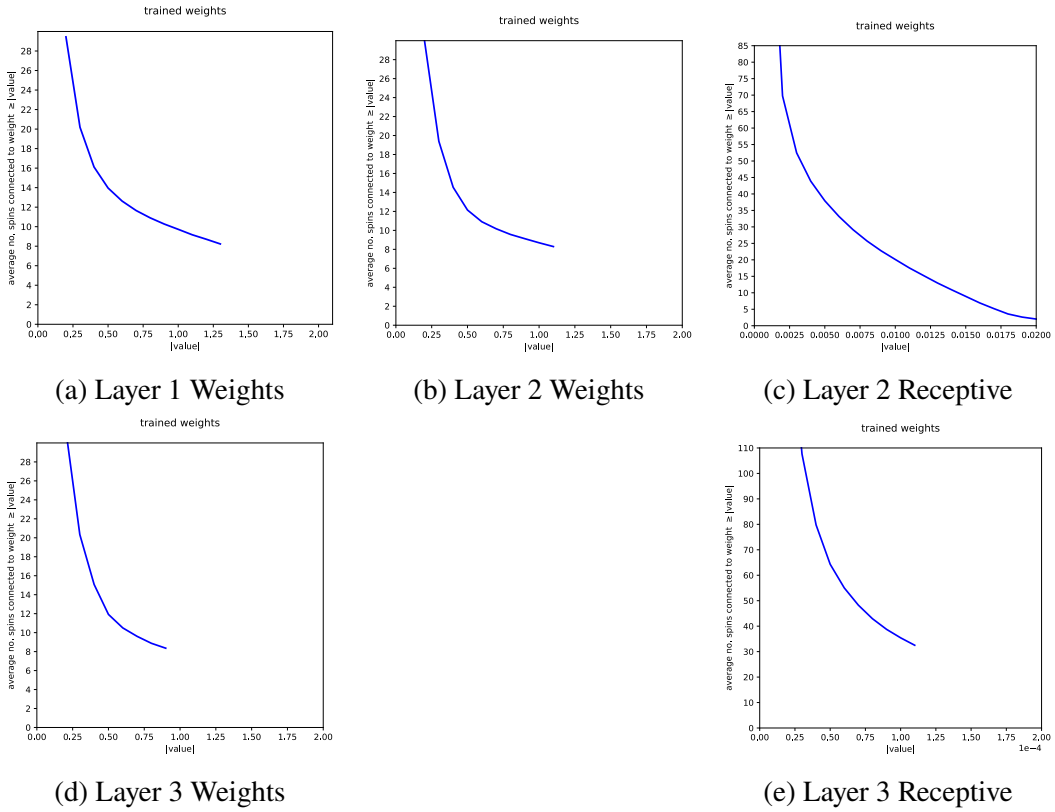
(a) Layer 1 Weights (b) Layer 2 Weights (c) Layer 2 Receptive

(d) Layer 3 Weights (e) Layer 3 Receptive

Figure 4.4: Average number of trained positive/negative weights with magnitude above a given value connected to a given spin location in input lattices. Results are given for both weight tensors and receptive field tensors for $\beta = .41$. The results suggest that we can think of each node in the RBM architecture being associated with two spins: a positive and negative.

From these results, we find that the structure of the weights and receptive tensors for all three layers are similar. (The receptive tensors have lower magnitude, but that's because there are more weights multiplied together). We find that we may approximate the weight tensor and receptive field plots as assigning only a single large positive weight and a single large negative weight to each spin, significantly lowering the number of relevant weight tensor components.

The results match for all β, as shown in Appendix A.2. This seems to suggest two things. First, it suggests that that models can be reconstructed using only the largest positive and negative weight, instead of the full weight tensor, which we examine in 4.5. Secondly, it further qualitatively connects the RBM's coarse-graining algorithm to that of 2D Ising model RG flow. Each node in the machine learning architecture can be thought qualitatively as corresponding to two spins: positive and negative, just as the renormalized Ising lattice "nodes" correspond to one spin. Thus, the



(a) Layer 1 Weights (b) Layer 2 Weights (c) Layer 2 Receptive

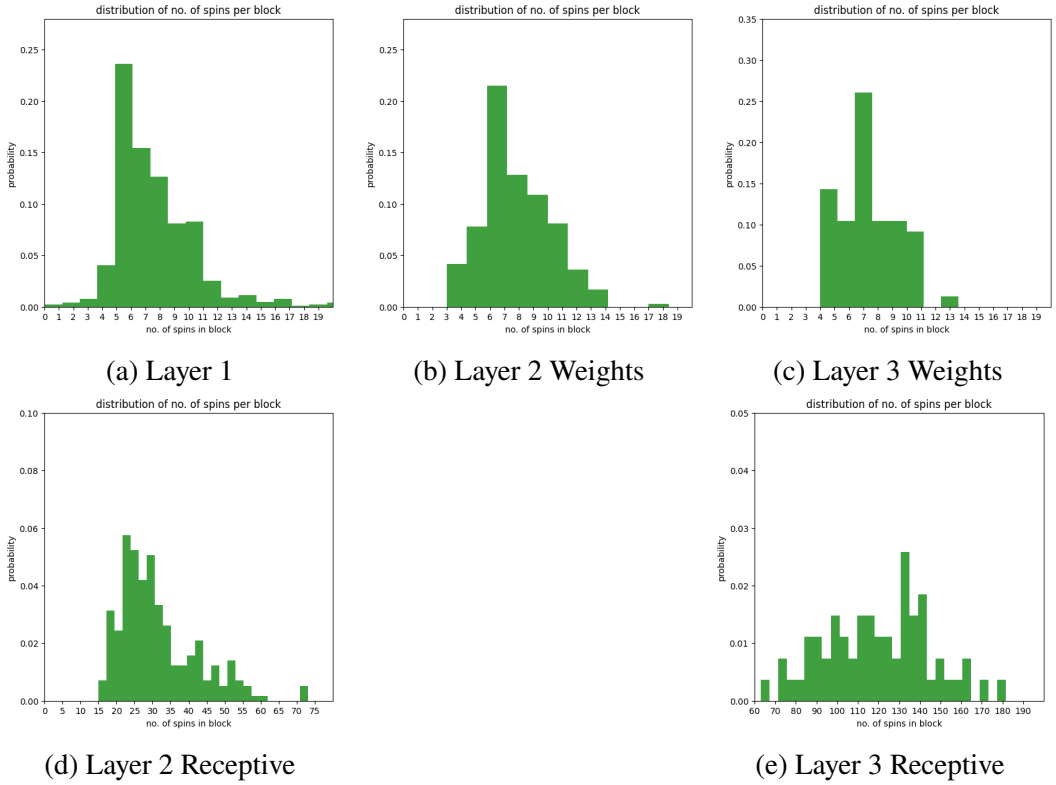
(d) Layer 3 Weights (e) Layer 3 Receptive

Figure 4.5: Average number of spin locations connected to each weight with magnitudes above a given value, such that all locations connect to at least two weights. Results are given for both weight tensors and receptive field tensors for $\beta = .41$. They show that we can approximate the RBM coarse-graining as a block spinning technique.

structure of the two systems are qualitatively much more similar than at first glance, with 2 weights per node instead of hundreds to thousands.

This viewpoint of the weights as spins allows us to do another analysis on the weight tensors. Now, instead of taking each Ising spin, and calculating how many weights are connected to each spin, we do the opposite: we take a magnitude and calculate how many spins are connected to a weight larger than that magnitude. These results are shown in Figure 4.5.

We find for both the weight tensors and the receptive field tensors that as we increase the magnitude of the weights, the number of spins connected to the weight drops off quickly. The higher magnitude weights are connected to only a few spins, forming "blocks". As discussed, these few high magnitude weights consist of the majority of the weights coupled to the RBM nodes. Thus, to some approximation, the RBM coarse-grains by creating blocks of nodes and connecting each block to two weights.



(a) Layer 1

(b) Layer 2 Weights

(c) Layer 3 Weights

(d) Layer 2 Receptive

(e) Layer 3 Receptive

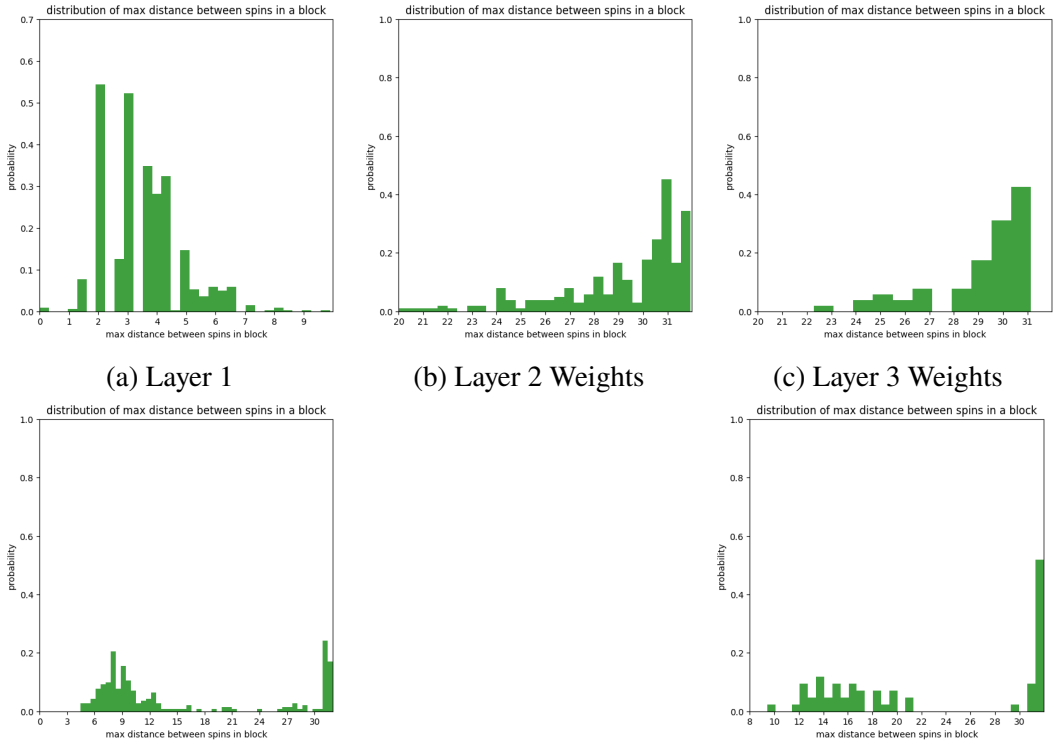
Figure 4.6: Histograms for the number of spins per block for all three layers' weight and receptive field tensors for $\beta = .41$. Results are highly consistent with Kadanoff $b = 2$ block spinning, as the blocks are about the size we'd expect for block spinning with two weights.

These results are consistent for all β, as shown in Appendix A.2.

4.4 Block Analysis

From the analysis in Section 4.3, we have shown that large weights in the RBMs only have a small number of nodes connected to them, forming blocks similar to those found in Kadanoff block spinning. In this section, we further examine the properties of these blocks to characterize the qualitative behavior of the model. We once again consider the properties of both the weight tensors and the receptive field tensors for all three layers.

To start, we plot the number of spins per block averaged over in each layer in 4.6. The results we find are quite similar to what we would expect from $b = 2$ block spinning. For the weight tensors of all three layers, we find that the block size peaks at slightly less than 8 spins per block. This is in line with what we expect for Kadanoff block spinning, in which there are 4 spins per block, and we multiply by 2 to account for both the positive and negative weights. The same result is seen in



(a) Layer 1

(b) Layer 2 Weights

(c) Layer 3 Weights

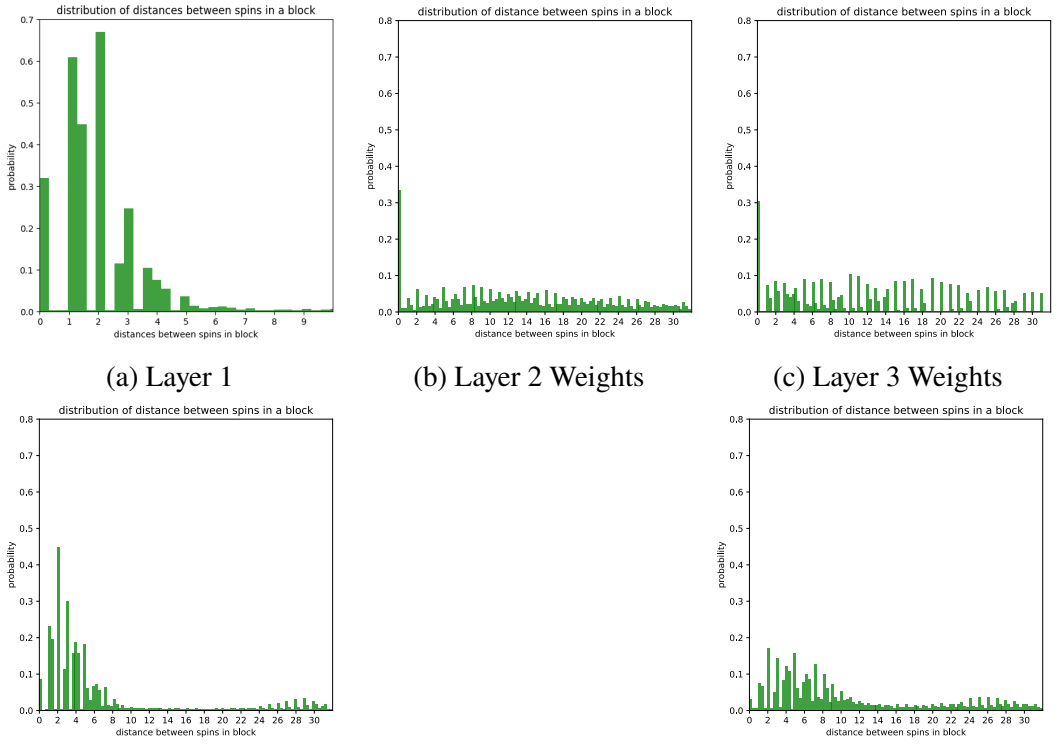
(d) Layer 2 Receptive

(e) Layer 3 Receptive

Figure 4.7: Histogram of average maximum distance per block for $\beta = .41$. Results are shown for both weight tensors and receptive field tensors. For layer 1, it seems that the system successfully learns locality. However, in layers 2 and 3, the individual weights contain nearly no information about locality, most blocks having a max 32 spins apart. The locality information is instead shown in the receptive field tensors, with about half of the layer 2 tensors and about a quarter of the layer 3 tensors learning locality.

the receptive field tensors, with the number of spins per block peaking slightly less than 32 for Layer 2 and around 128 for Layer 3. This is consistent with the blocks of 16 and 64 expected from block renormalization, once again multiplied by 2 for both signs of spin. These results are consistent for all β, as shown in Appendix A.3.

From here, we examine the distances between spins in the blocks. We start by plotting the maximum distance between spins in a block in Figure 4.7. We find that for the weight tensor in layer 1, the distribution peaks around a maximum distance of 3 spins between spins in a block. These short range distances imply that the blocks in layer 1 are formed from short range correlations, as expected in the Ising model. We find that this pattern breaks for the weight tensors in layers 2 and 3, with the maximum distance between spins in blocks almost always reaching its maximum of 32 spins. This implies that the weight tensors themselves do not contain information



(a) Layer 1 (b) Layer 2 Weights (c) Layer 3 Weights

(d) Layer 2 Receptive (e) Layer 3 Receptive

Figure 4.8: Histogram of distance between all spins, averaged over all lattices. We analyze both the weight tensors and receptive field tensors. We find that locality is mostly learned for all three tensors, with most of the receptive tensors being close together, and only a couple far away. This is not true for the individual weights, which do not learn locality for layers 2 and 3.

about Ising locality, as blocks hold over long distances. The receptive field tensors, however, do contain some information about locality, with about half the layer 2 tensors learning some form of locality (with a distance of about 7 to 8 between spins), and a quarter of the layer 3 tensors learning locality (with a distance of about 12 to 16 between spins).

We can further examine the structure of our block spinning via examining all distances between spins, instead of just the maximums. These results are shown in Figure 4.8. We find similar results to our previous analysis. In particular, we find that the weight tensor for layer 1 learns locality that peaks with an average distance of 2 between nearby spins. We also find that the layer 2 and 3 weight tensors contain no indication of such locality, almost uniformly distributed throughout the distances. The receptive field tensors contain some information about locality, with the vast majority of the distances between spins in layer 2 and layer 3 being lower than 10

distances apart.

These results suggest that while the spin size of the blocks from the RBM's are highly consistent with $b = 2$ renormalization group flow, they also slightly struggle with learning locality. The results also imply that weight tensors from layers 2 and 3 do not characterize the system well, but the receptive field tensors characterize the system better, as expected from our receptive field plots. We find from our results that the receptive field tensor for layer 1 contains blocks that are entirely local (only 6 spins wide). On the other hand, layers 2 and 3 contain blocks in which most spins are close to each other with a few spins up to 32 spins away.

We further note from plots in Appendix A.3 that the average distance between spins typically does not change as β changes, though as β reaches the critical point less and less locality is learned (likely due to diverging correlation lengths). This lack of peak change is different than what we'd expect from direct comparison of the weights to Ising model spins, due to correlation length changing with temperature. Even though block spinning is a useful model for the RBM, this suggests a deeper numerical analysis is needed to continue to connect the two concepts past this qualitative level. This numerical analysis is discussed in Chapter 5.

4.5 Model Reconstructions

In addition to analyzing the weights of the model, we also consider how it works as an auto-encoder to reconstruct the training data. To do so, we consider the hidden weight probabilities $\text{pHid}_{i_n j_n}$ as defined in 1.13, where n is the layer number, to create a reconstructed lattice

$$\text{vReco}_{ij}^{Ln} = \sum_{i_n, j_n} \sigma(\text{rec}_{ij;i_n j_n}^{Ln} \cdot \text{pHid}_{i_n j_n}), \tag{4.4}$$

where σ is Equation 1.12. We also consider the results of replacing $\text{rec}_{ij;i_n j}$ in Equation 4.4 with just the two large weights. These results are shown in Figure 4.9 with the first row consisting of the original lattices, the next three rows consisting of the reconstructions from each layer in numerical order, and the last three rows consisting of the reconstructions with just using the large weights.

We find that the reconstructions are quite good for $\beta = .41$, with just the finer detail disappearing as we go down the layers, reproducing a result from [16]. We also find that by only including the large positive weights, the reconstructions are slightly better than the ones without them, keeping together some of the finer detail lost in the other reconstructions. These results are qualitatively consistent with the idea that

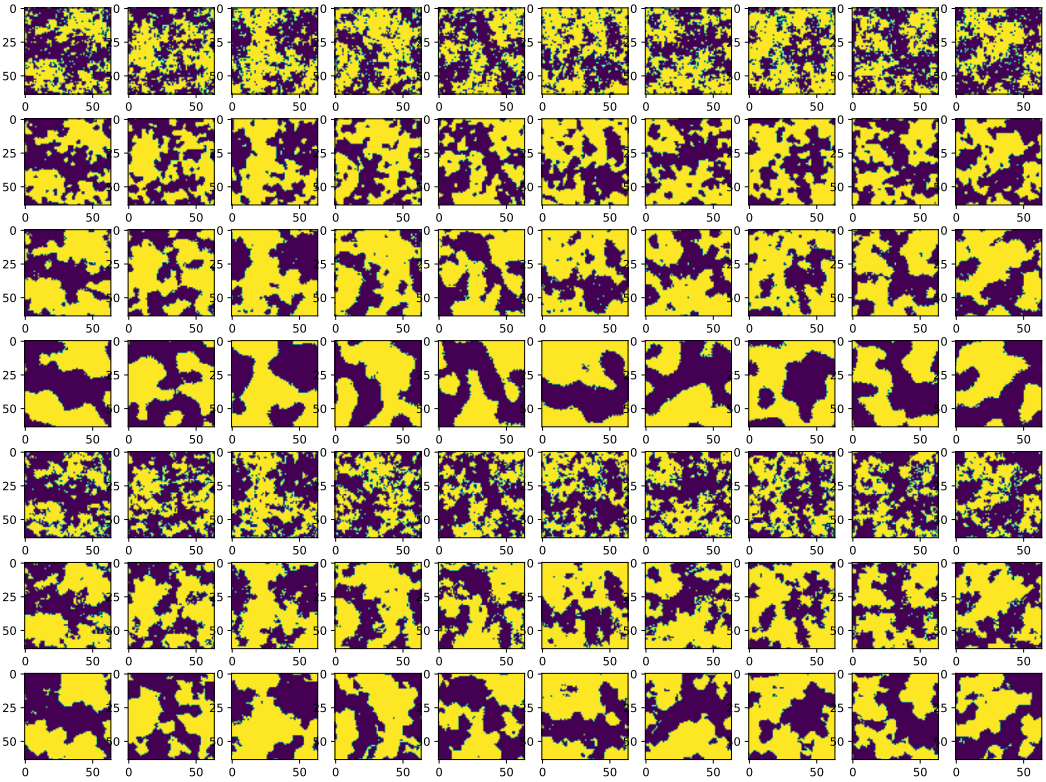


Figure 4.9: Reconstructed model plot for $\beta = .41$. The first row consisting of the original lattices, the next three rows consisting of the reconstructions from each layer in numerical order, and the last three rows consisting of the reconstructions with just using the large weights. The reconstructions are accurate, with the ones using only the large weights more accurate.

the majority of the RBM's information in stored in only two weights. In addition, it is consistent with the idea that the RBM coarse-grains in a similar manner to RG flow, keeping the data's macroscopic structure, but not its the microscopic structure.

However, these results are not consistent for all values of β. We find from plots in Appendix A.4 that the reconstructions get worse as we approach criticality, suggesting that the network fails as an autoencoder in these cases, likely due to the excess of one spin over another. However, the large weight reconstructions are more consistent across all the β values (though they still get worse near criticality). This implies that [16]'s argument that the RBM's make a good autoencoder does not apply as one gets closer to criticality, meaning the RG flow connection may fail at these temperatures. However, it also implies that most of the important information in the RBM is encoded within the two large weights, giving us an interesting view on the structure of the RBM learning.

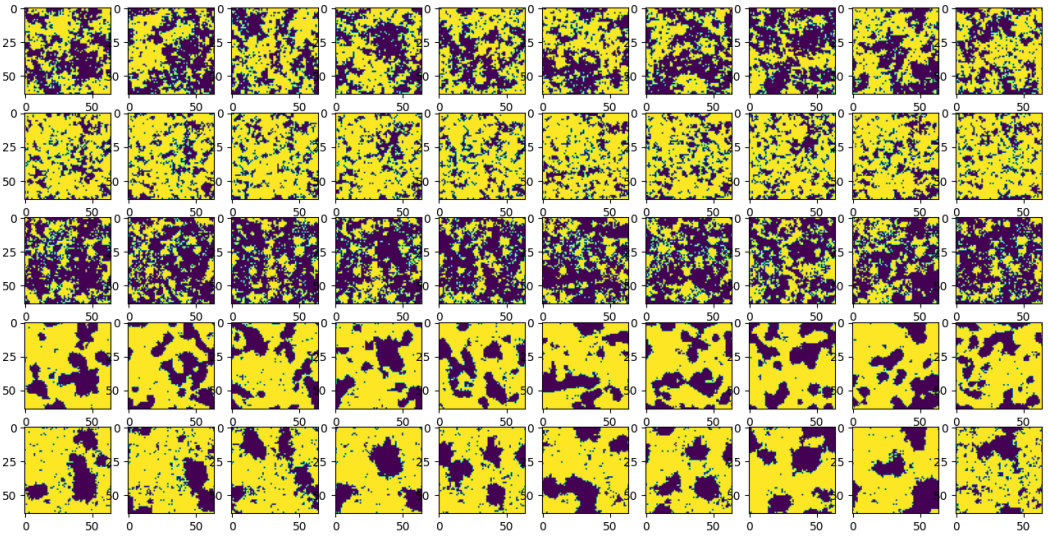


Figure 4.10: Another reconstructed model plot for $\beta = .41$. We consider the case in which we only use large weights to calculate the hidden spins and reconstructed spins, shown in row 2. Row 1 consists of original lattices while rows 3-5 consist of the reconstructed lattices using full weights for all three layers. The reconstruction that changes the hidden weights failed, suggesting that the full weight tensors are needed in the calculation of hidden weights, instead of just two large weights.

We also attempted to reconstruct the model by using only the two large weights to create the hidden weights used in calculating $pHid_{i_n j_n}$, along with using the two large weights in the receptive field tensor. We did this only for the first layer at $\beta = .41$. These results are shown in Figure 4.10, where the first row is the original lattice, the second row is the reconstruction with only the large weights, and the last three row are the reconstructions from all three layers for comparison. Obviously, from the figure, the reconstruction where we changed the hidden weights to only consider large weights does not work well, implying that the full weight tensor is needed when creating the hidden weights. In this sense, the smaller weights have an effect on the outcome of the model: they are needed to get correct hidden weights. We thus conclude that the two weight model is a good approximation of how the RBM learns, but is nowhere near the full extent of the RBM learning. We did not run this test for other layers or other values of β because of the drastic failure of the reconstruction.

Chapter 5

CONNECTING RENORMALIZATION TECHNIQUES AND DEEP LEARNING

In the previous sections, we discussed the renormalization group and machine learning separately, focusing on the validation of our renormalization techniques in Chapters 2 and 3, and a qualitative analysis of the deep learning of RBMs in Chapter 4. It is now time to put the two concepts together by creating direct Ising representations from the RBMs based on our qualitative analysis, feeding them into the renormalization group validation, and comparing results. We do this in two different ways: running the learning in reverse and generating 64 by 64 lattices from renormalized 8 by 8 lattices, and creating a direct Ising spin representation from our RBM weights.

5.1 Generative Models

In examining the quantitative success of the deep learning infrastructure as a renormalization group flow, we first used the learning weights as a way to generate new 64 by 64 lattice models from 8 by 8 lattice models. We then compared these generated models to those derived from the Wolff algorithm.

To do so, for each value of β in the 64 by 64 lattice, we find the 8 by 8 β value it flows to in the RG flow in Chapter 3. These results are shown in Table 5.1. We then generate 8 by 8 lattices for each of these 8 by 8 β values using the Wolff algorithm,

64x64 β Value	8x8 β Value
.395	.241
.4	.256
.405	.274
.41	.291
.415	.315
.42	.338
.425	.367
.43	.397

Table 5.1: β values for the 64 by 64 Ising modes and the corresponding β values for the 8 by 8 Ising models the original values flow to. The 8 by 8 Ising models are used to generate new 64 by 64 models by running the learning backwards.

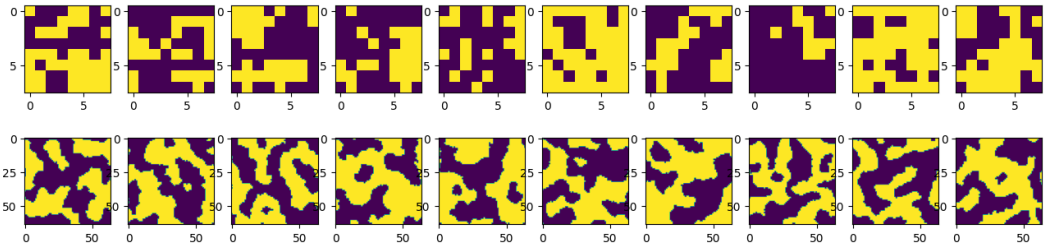


Figure 5.1: Generative model for $\beta = .41$. The first row contains the original 8 by 8 lattice models and the second row contains the 64 by 64 generated model.

which we denote here as $x^{L3}_{i_3 j_3}$. From here, we generate the model by taking:

$$x^{L2}_{i_2 j_2} = 2\sigma\left(\sum_{i_3, j_3} w^{L3}_{i_2 j_2; i_3 j_3} \cdot x_{i_3 j_3}\right) - 1, \tag{5.1}$$

then

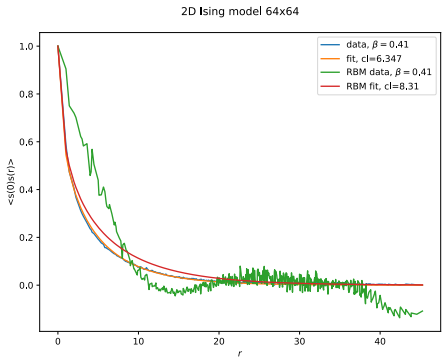
$$x^{L1}_{i_1 j_1} = 2\sigma\left(\sum_{i_2, j_2} w^{L2}_{i_1 j_1; i_2 j_2} \cdot x_{i_2 j_2}\right) - 1, \tag{5.2}$$

and lastly,

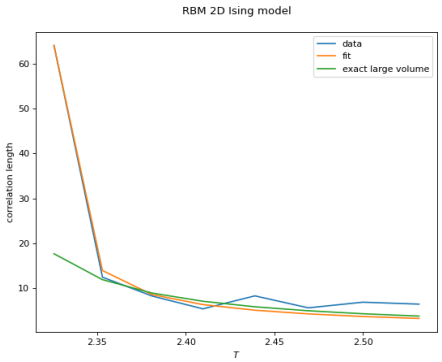
$$v^{gen}_{ij} = 2\sigma\left(\sum_{i_1, j_1} w^{L1}_{ij; i_1 j_1} \cdot x_{i_1 j_1}\right) - 1. \tag{5.3}$$

We do this for 20,000 lattices for all β. The result of this procedure for $\beta = .41$ is shown in Figure 5.1. The first row contains the original 8 by 8 lattice models and the second row contains the 64 by 64 generated models. At first glance, there seems to be little relation between the original lattices and the generated model, seeming to throw out the idea of the connection. However, this is not a problem. The RBM takes in our lattices as one dimensional and uses this information to rederive the spin connections. Due to this, the spin blocks get "mixed-up" and moved to different locations, and sometimes all the blocks get flipped. We deal with this more effectively in Section 5.2. However, the generative model results are still useful, as the quantitative information, including the correlations, stays the same. This means that these generative models should accurately reflect the Ising model quantitatively. These general results hold for all β, as shown in Appendix B.1.

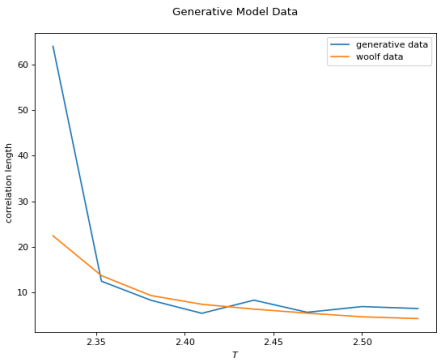
After doing this for all of our values of β, we proceed with numerically analyzing our results, in the same manner that we analyzed the Wolff algorithm in Chapter 3. Some results of this analysis are shown in Figure 5.2, including the correlation length fit for



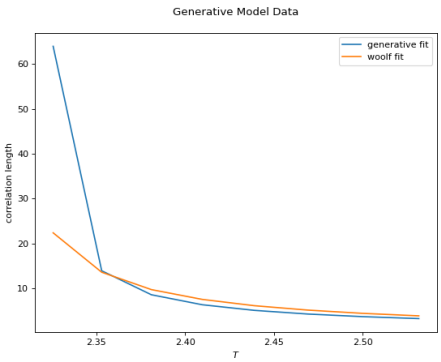
(a) Correlation length for $\beta = .41$.



(b) Generative Model



(c) Generative Model and Wolff Algorithm Comparison



(d) Generative Model and Wolff Algorithm Comparison

Figure 5.2: Comparing the Wolff algorithm with the generative model. The generative model tends to have a higher correlation function than expected for $0 < r < 10$ and a lower correlation function for $10 < r < 20$, for all β. For the correlation length temperature dependence, we find that $\nu = 0.763$ for the generative model, as opposed to the Wolff algorithm value of $\nu = 0.973$. The generative model only somewhat matches the fit at $T > 2.35$, and does not for $T < 2.35$.

$\beta = .41$ for both the Wolff algorithm and generative models, the generative model correlation length data and fits, and comparisons between the generative model and Wolff model. The rest of the correlation length fits are shown in Appendix B.1.

We find that the correlation length fits to the generative models tend to have higher correlation functions than expected for $0 < r < 10$ and a lower correlation function for $10 < r < 20$. This suggests that the reproductions do not perfectly match that of the Ising model, and on some level, the RBM fails to reproduce the Ising model due to these offsets. However, we still proceed with the analysis assuming that these over-estimations and under-estimations cancel out.

When we calculate the temperature dependence of these correlation lengths, we find

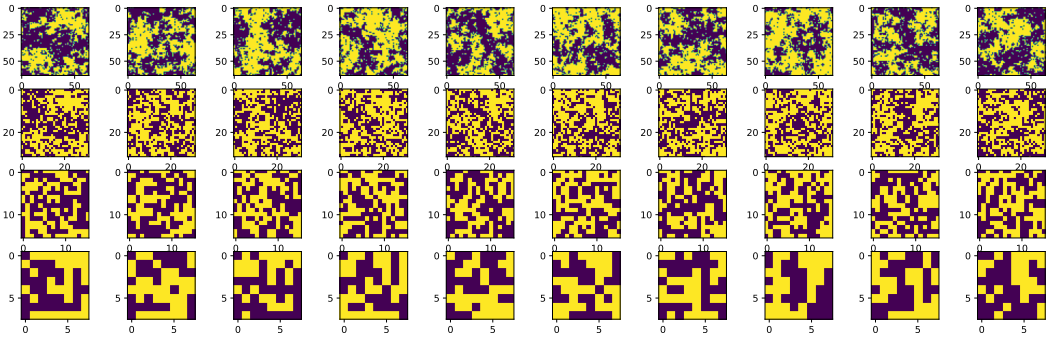


Figure 5.3: RBM coarse graining for $\beta = .41$. The first row consists of the original Wolff-generated Ising lattices and the next three rows consist of Ising spin representations of the RBM learning layers. We find that the first two layers are qualitatively similar to the renormalization group.

that the data only somewhat matches the fit at $T > 2.35$, and does not for the data point at $T < 2.35$. The fit suggests that $\nu = 0.763$ for the generative model, as opposed to the Wolff algorithm value of $\nu = 0.973$. These results imply that the generative models forms a poor approximation of the nearest-neighbor Ising model, with a few qualitative connections (such as the fit going in the correct direction), but little quantitative connections.

While this data weakens the idea of a naive connection between the learning and the renormalization group, it does not rule out the connection completely. The models here were generating assuming that the group flow could be approximated using only the nearest-neighbor Ising models and only the coupling β. To further rule out a quantitative connection between RBM generative models and RG flow, this extra parameter space must be explored.

5.2 From Weights to Ising Spins

Using generative models, however, is not the only way to connect the RBM learning to the Ising spins. Instead, we can sample from our probability distributions pHid from Equation 1.13 for each layer, and then convert the binary spins of those layers to spins of ± 1.

As stated in Section 5.1, the locations of these new lattices do not have a simple relationship to the locations of the original lattices. We fix this by taking the receptive field tensors $\text{rec}^{Ln}_{ij;i_n j_n}$ and finding which of the entries i, j have the largest absolute value for each of the i_n, j_n. This is the location of the original spin that the weight is most connected to. We then coarse grain and define this as the location of the corresponding spin.

The results of this coarse graining procedure for $\beta = .41$ are given in Figure 5.3. The first row consists of the original Wolff lattices, while the next 3 rows consist of the 3 layers of coarse graining in order. We do this for all β as shown in Appendix B.2.

The results here qualitatively show a coarse graining that is similar to Kadanoff-block spinning in some ways, with the large scale structure of the system being most strongly preserved in the second layer. The first layer of the coarse graining still preserves large scale structure, but not as well as the second layer, having extra noise. The last layer of the coarse graining, however, loses most of the macroscopic structure. This result is similar throughout all β values. This strengthens the idea of a qualitative connection between the renormalization group and deep learning in the first two layers, while the third layer loses this qualitative connection.

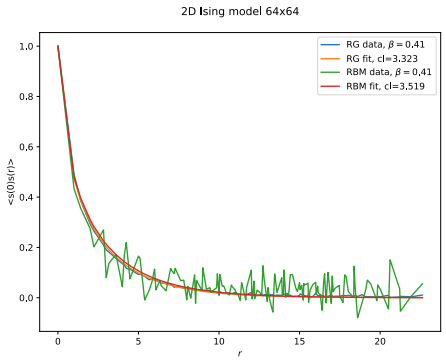
We further examine this quantitatively by running the RBM coarse-grained models through the validation procedure from Chapter 3, and comparing these results with both our traditional renormalization results and the Wolff algorithm. In our plots, we also include a correlation length fit for $\beta = .41$ for reference. The remaining correlation length plots are given in Appendix B.2.

For layer 1, our results are shown in Figure 5.4. We have that most of our correlation function data matches the fits decently well, with the exception of a couple of points, like $\beta = .415$ and $\beta = .43$. In particular, when we fit the data, we get $\nu = 1.178$ for the Wolff algorithm, $\nu = 0.983$ for the RG flow and $\nu = 1.174$ for the RBM training. Looking only at the fits, the three cases seem to match quantitatively!

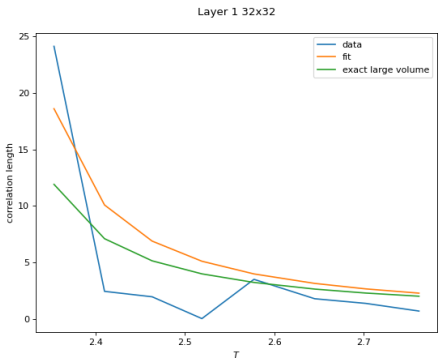
However, the RBM data does not match up with its fit very well. The overestimates and underestimates cancel to make the fit work, but the data and fit mismatch prevent us from concluding the first layer as qualitatively equivalent to the renormalization group results.

For layer 2, our results are shown in Figure 5.5. As in layer 1, the correlation lengths tend to match their fitting function, with the exception of $\beta = .415$. When we fit the correlation length temperature dependence, we found the value of $\nu = 1.452$ for the Wolff algorithm, $\nu = 0.969$ for block spinning, and $\nu = 0.729$ for the RBM training. All three fits are close together above criticality, but diverge from one another as criticality is approached. This causes some quantitative mismatching.

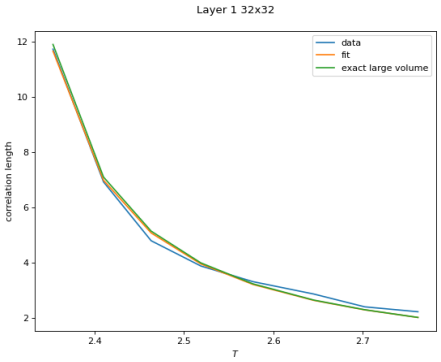
Additionally, the same issue of a mismatch between data and fit appears here. The fit we use is rather restrictive, and our data undershoots it. The renormalization does



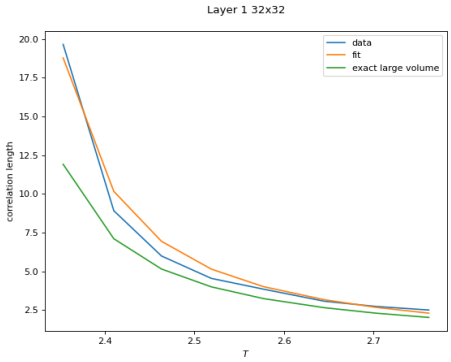
(a) Correlation length comparison for $\beta = .41$.



(b) RBM Results



(c) RG Blocks

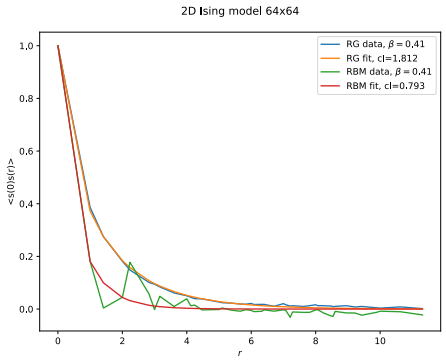


(d) Wolff Algorithm

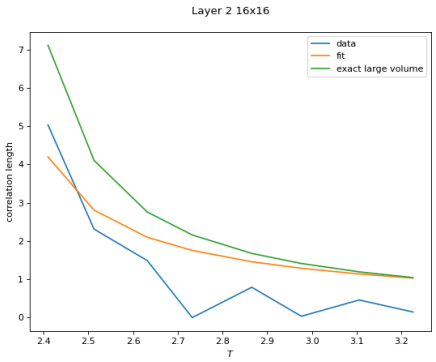
Figure 5.4: Analysis of layer 1 RBM Ising representation. The fits for the RBM correlation function are consistent with the data for most β. For the correlation length temperature dependence, we find that $\nu = 1.178$ for the Wolff algorithm, $\nu = 0.983$ for the RG flow and $\nu = 1.174$ for the RBM training. The RBM data, however, does not match this fit well.

appear to be going in the right direction, qualitatively, however.

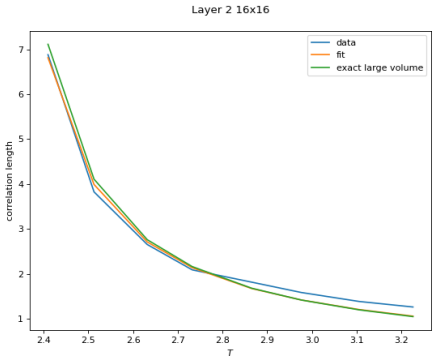
For layer 3, our results are shown in Figure 5.6. Here, even the qualitative results seem to change. Fits of the correlation function start being more inconsistent with the fits, and correlation lengths are all close to zero. For temperature dependence, we get $\nu = 0.972$ for the RG flow, $\nu = 1.933$ for the Wolff algorithm and $\nu = 0.0$ for the RBM. Here, even the qualitative results suggesting that the RBM is connected to RG completely fall apart, reinforcing the earlier qualitative conclusion we had from looking at the coarse-graining directly. We also have discrepancies near criticality for the Woolf algorithm and RG flow, suggesting that the 8 by 8 model may be too small to effectively do work with.



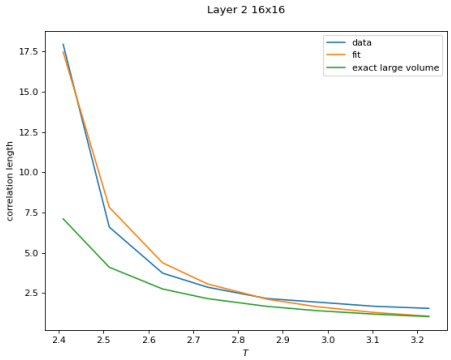
(a) Correlation length for $\beta = .41$.



(b) RBM Results

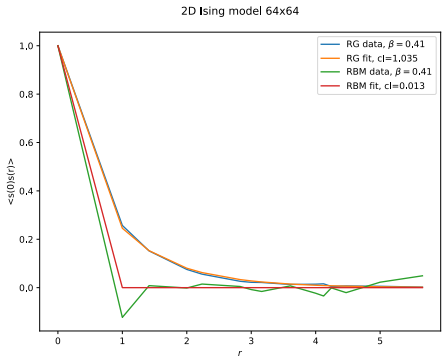


(c) RG Blocks

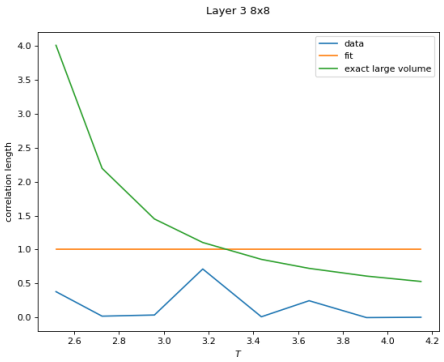


(d) Wolff Algorithm

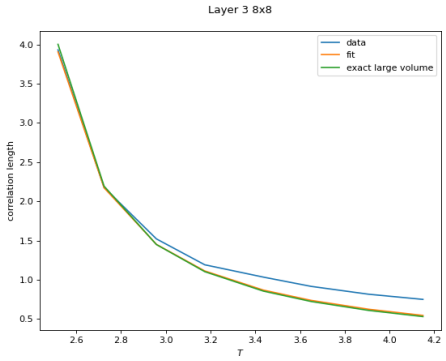
Figure 5.5: Analysis of layer 2 RBM Ising representation. The fits for the RBM correlation function are consistent with the data for most β. For the correlation length temperature dependence, we find that $\nu = 1.452$ for the Wolff algorithm, $\nu = 0.969$ for block spinning, and $\nu = 0.729$ for the RBM training. The RBM data does not match this fit well.



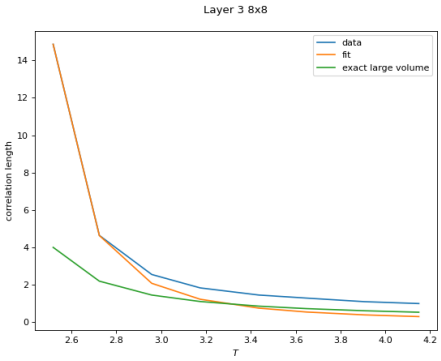
(a) Correlation length for $\beta = .41$.



(b) RBM Results



(c) RG Blocks



(d) Wolff Algorithm

Figure 5.6: Analysis of layer 3 RBM Ising representation. The fits for the RBM correlation function are often inconsistent with the data and correlation lengths are close to zero. For the correlation length temperature dependence, we find that $\nu = 0.972$ for the RG flow, $\nu = 1.933$ for the Wolff algorithm and $\nu = 0.0$ for the RBM. The RBM data is not numerically consistent with RG flow at all for this layer.

Chapter 6

CONCLUSIONS AND FURTHER WORK

6.1 General Conclusions

Throughout this work, we have considered applications of basic machine learning techniques to renormalization in the one dimensional and two dimensional Ising models (in Chapters 2 and 3), discussed the qualitative results of deep learning using RBMs of the the two dimensional Ising model (in Chapter 4), and attempted to connect the two paradigms quantitatively (in Chapter 5). What, then, can we conclude?

Our first conclusion is that machine learning techniques through Adam optimization can be successfully applied in renormalizing both the 1D and 2D Ising models. These optimization techniques work best when using correlation lengths as loss functions, and when paying close attention to how finiteness affects criticality. We found better renormalization results using optimization techniques than we did using accepted analytical techniques, like the Maris-Kadanoff equation.

Our second conclusion is that there are strong qualitative connections between deep learning through RBMs and RG flow, just as Mehta and Schwab argued in [16]. This is shown through the reproduction of layer 1 and 3 receptive field plots and model reconstruction for 8 β values. In addition, we discovered the emergence of a two-weight blocking structure similar to the Ising model one-weight blocking structure, and found that most of these blocks contain some sense of locality. Furthermore, when converting these blocks to Ising spin representations, layer 1 and 2 had correlation function structures that match how they should look in the Ising model. In addition, the layer 1 fit ν value is very close to its expected value, and layer 2 correlation lengths tend to go in the correct direction for renormalization.

Our third conclusion, however, is that we do not have enough evidence to rule in favor of this connection existing on a quantitative level. In particular, the layer 3 model failed to produce results that looked like the Ising model, even qualitatively, with $\nu = 0$. Similarly, the generative model had trouble with deriving the correlation lengths due to fitting issues. Models for layers 1 and 2 had data that did not necessarily correspond to their fits, preventing us from establishing a qualitative connection there. Only 8 data points were used, some with poor reconstructions,

and we have yet to examine the case of RBMs without L1 regularization.

That being said, these quantitative discrepancies do not imply that the connection is non-existent either. As discussed in Chapter 3, the nearest-neighbor renormalization is just a subspace of the total parameter space. It is possible that our results here tell only part of the story because we only accomplish the group flow in β. This is discussed more in the next section.

6.2 Future Quantitative Work

To make a more solid conclusion about the quantitative connection between the renormalization group and deep learning, further developments need to be made on both renormalization group and deep learning structures.

For the renormalization group, extending the group flow to be a two parameter flow using the next-to-nearest neighbor Hamiltonian would provide more parameter space to examine the RBM flow in, getting a clearer picture on why the numerical connections are not present in the 1D space. In addition, criticality issues from using finite systems likely also played a role in the quantitative disconnect. Using larger systems should help prevent these criticality issues.

In addition, the layer 3 failure could have just been an issue of the 8 by 8 system being too small. Changing the entire system to start at 128 by 128 or 256 by 256 would allow us to ignore this as a possible problem. In addition, running the numerical analysis for more β values would give us clearer data points on the connection between the two, and make our fits clearer.

Lastly, our results here only considered what happens when we use an L1 penalty for learning. Removing this penalty would allow us to see how much the penalty affects how clearly and quickly the system is learning locality.

6.3 Further Extensions

If these numerical issues are resolved and the connection made clearer, one could extend the work further using other renormalization models and more modern machine learning methods.

In particular, one could extend the work from the 2D Ising model to the Ising model in 3 or more dimensions. Similarly, one could look at Ising variations, such as the quantum Ising model or the tricritical Ising model. One could also extend it to other O(N) models, like the XY or Heisenberg models.

Similarly, one could use more modern deep learning techniques, instead of the simple case of RBMs. One example in particular could be the use of variational autoencoders.

All in all, though the examination of a possible connection between machine learning and renormalization has been greatly expanded on in our work here, there still remains plenty of work to be done to fully understand the extent of such a connection.

Bibliography

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016. URL https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

[2] Enrique Alvarez-Lacalle, Blas Echebarria, Jon Spalding, and Yohannes Shiferaw. Calcium alternans is due to an order-disorder phase transition in cardiac cells. *Phys. Rev. Lett.*, 114:108101, Mar 2015. doi: 10.1103/PhysRevLett.114.108101. URL https://link.aps.org/doi/10.1103/PhysRevLett.114.108101.

[3] Serena Bradde and William Bialek. Pca meets rg. *Journal of Statistical Physics*, 167(3–4):462–475, Mar 2017. ISSN 1572-9613. doi: 10.1007/s10955-017-1770-6. URL http://dx.doi.org/10.1007/s10955-017-1770-6.

[4] Thomas Bury. Statistical pairwise interaction model of stock market. *The European Physical Journal B*, 86(3), Mar 2013. ISSN 1434-6036. doi: 10.1140/epjb/e2013-30598-1. URL http://dx.doi.org/10.1140/epjb/e2013-30598-1.

[5] John Cardy. *Scaling and Renormalization in Statistical Physics*. Cambridge Lecture Notes in Physics. Cambridge University Press, 1996. doi: 10.1017/CBO9781316036440.

[6] Miguel Á. Carreira-Perpiñán and Geoffrey E. Hinton. On contrastive divergence learning. In *International Conference on Artificial Intelligence and Statistics*, 2005.

[7] Johanna Erdmenger, Kevin T. Grosvenor, and Ro Jefferson. Towards quantifying information flows: relative entropy in deep neural networks and the renormalization group, 2021. URL https://arxiv.org/abs/2107.06898.

[8] Geoffrey Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14:1771–800, 09 2002. doi: 10.1162/089976602760128018.

[9] Geoffrey E. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*, pages 599–619. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8. doi: 10.1007/978-3-642-35289-8_32. URL https://doi.org/10.1007/978-3-642-35289-8_32.

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[12] Eloi-Alain Kyimba. *Comparison of Monte Carlo Metropolis, Swendsen-Wang, and Wolff Algorithms in the Critical Region for the 2-dimensional Ising Model*. PhD thesis, 3AD.

[13] Henry W. Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, Jul 2017. ISSN 1572-9613. doi: 10.1007/s10955-017-1836-5. URL http://dx.doi.org/10.1007/s10955-017-1836-5.

[14] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, Zihao Wu, Dajiang Zhu, Xiang Li, Ning Qiang, Dingang Shen, Tianming Liu, and Bao Ge. Summary of chatgpt/gpt-4 research and perspective towards the future of large language models, 2023.

[15] Joseph Lykken. Physic theory and deep learning. slides, 2017.

[16] Pankaj Mehta and David J. Schwab. An exact mapping between the variational renormalization group and deep learning, 2014.

[17] Arnab Paul and Suresh Venkatasubramanian. Why does deep learning work? - a perspective from group theory, 2015.

[18] Sebastian Ruder. An overview of gradient descent optimization algorithms, 2017.

[19] David J. Schwab and Pankaj Mehta. Comment on "why does deep and cheap learning work so well?" [arxiv:1608.08225], 2016.

[20] Nathan Zhao. Cluster monte carlo. https://github.com/zhaonat/cluster_monte_carlo, 2018.

Appendix A

TWO DIMENSIONAL ISING LAYER-BY-LAYER ANALYSIS PLOTS

A.1 Receptive Field Plots

Each individual plot represents a node in a given layer and the values on the plot represent the full set of receptive field tensors corresponding to that node. The clustering of non-zero tensors next to each other suggests the model is qualitatively learning some form of locality. This is discussed in depth in Section 4.2.

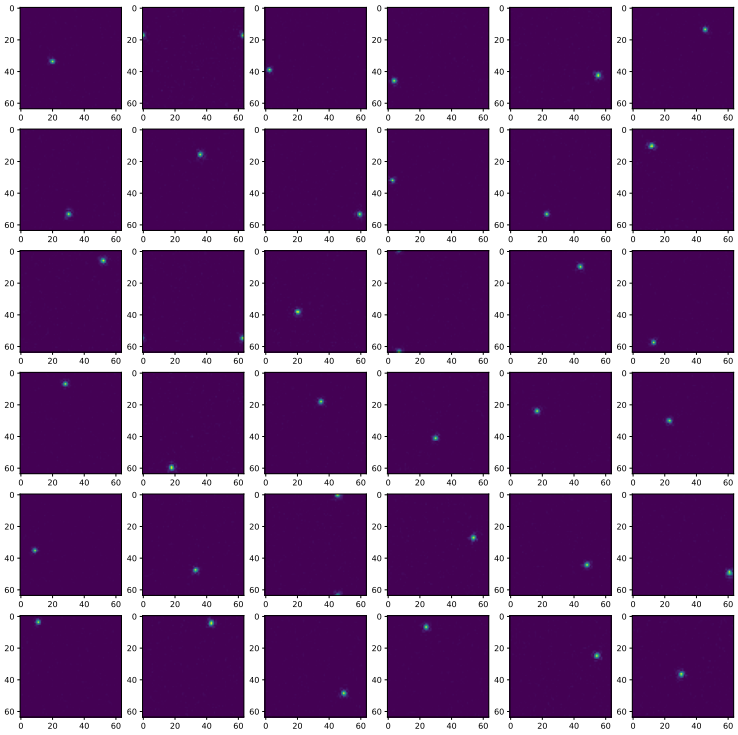


Figure A.1: Layer 1 receptive field plot for $\beta = .395$.

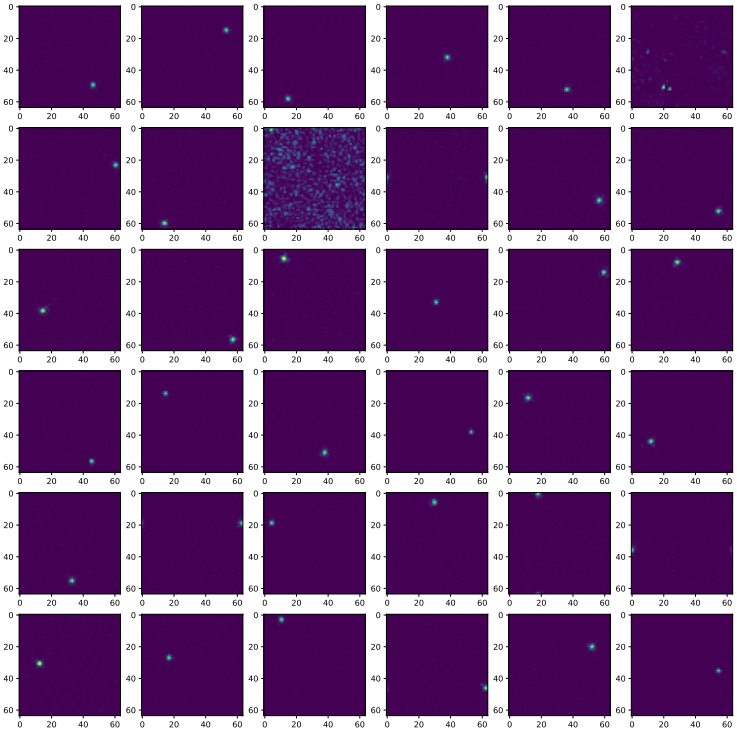


Figure A.2: Layer 1 receptive field plot for $\beta = .4$.

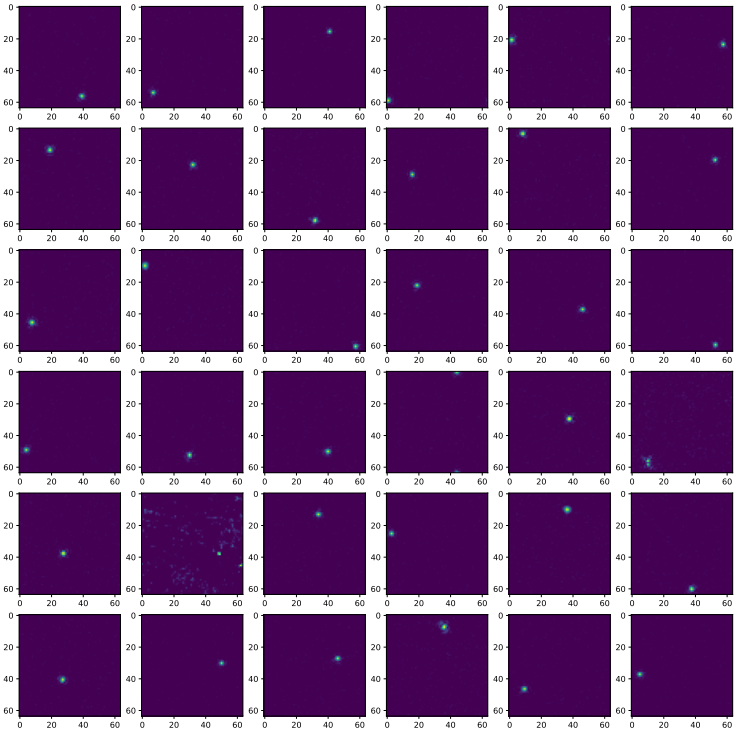


Figure A.3: Layer 1 receptive field plot for $\beta = .405$.

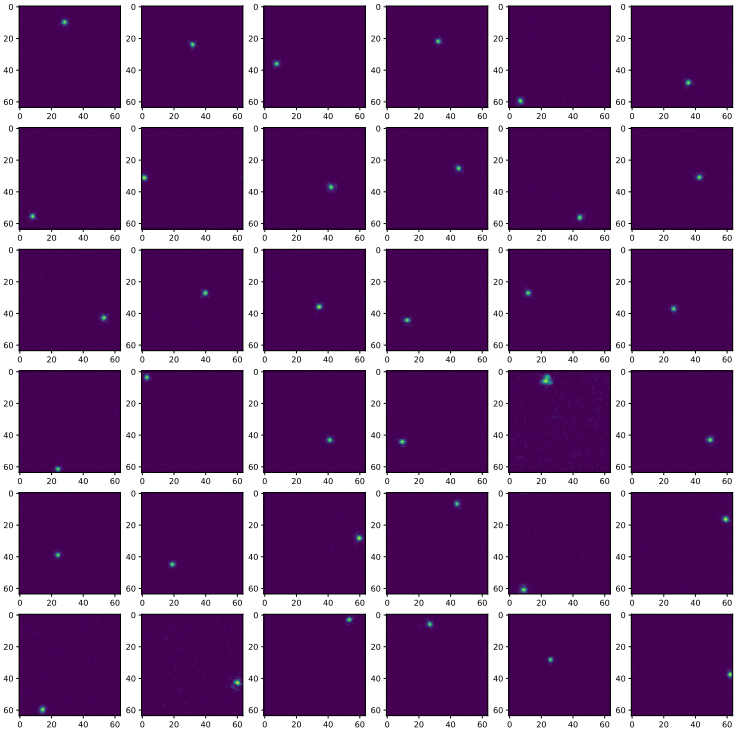


Figure A.4: Layer 1 receptive field plot for $\beta = .41$.

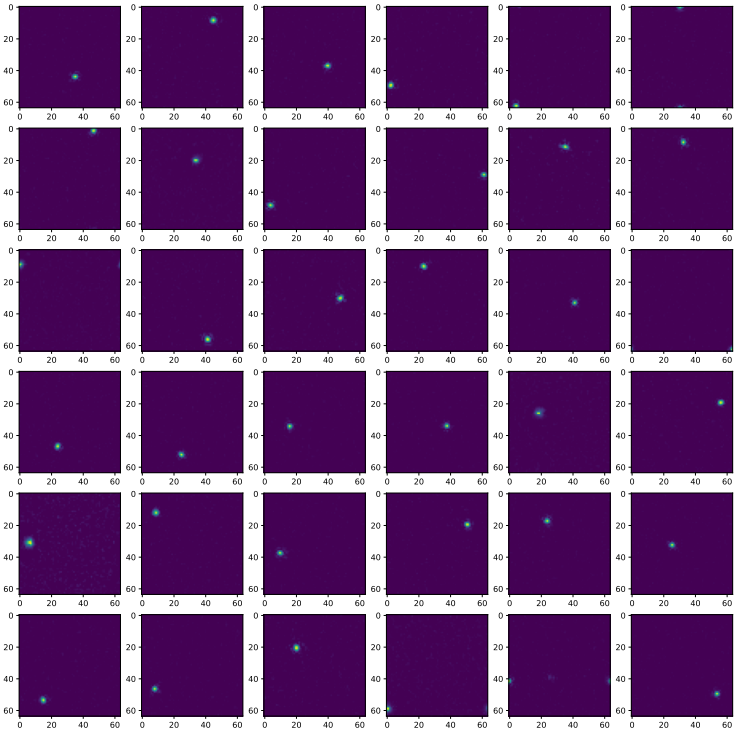


Figure A.5: Layer 1 receptive field plot for $\beta = .415$.

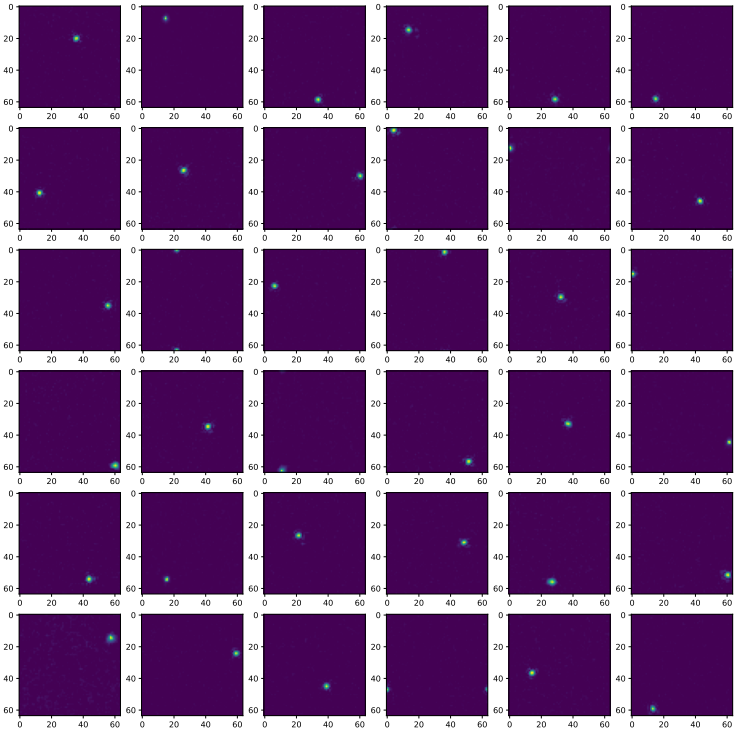


Figure A.6: Layer 1 receptive field plot for $\beta = .42$.

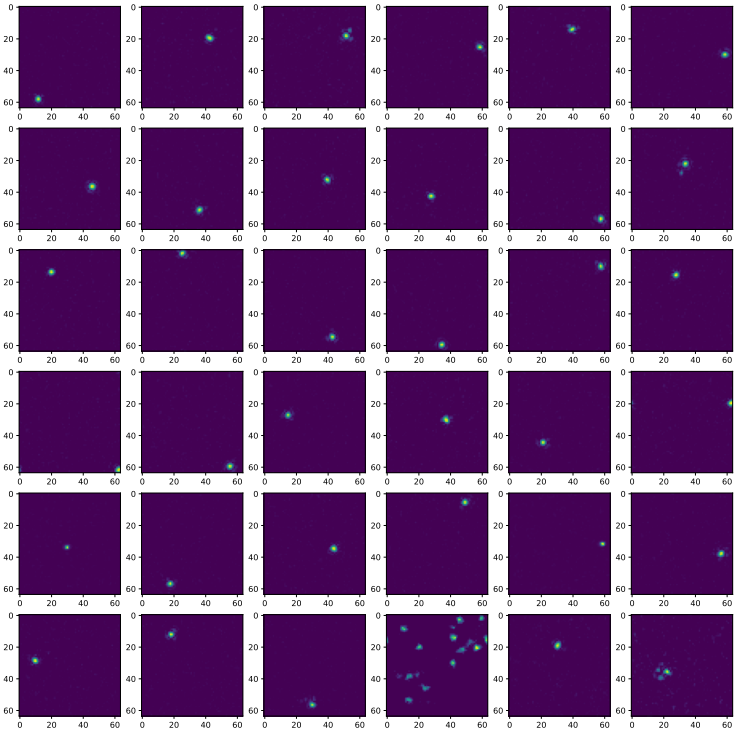


Figure A.7: Layer 1 receptive field plot for $\beta = .425$.

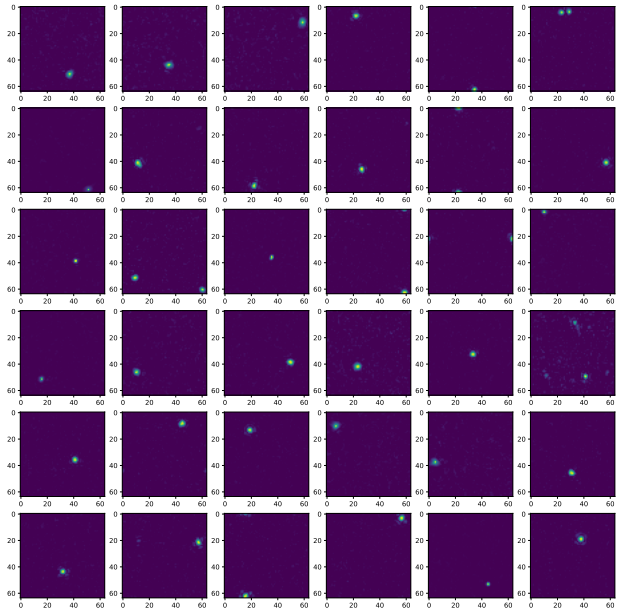


Figure A.8: Layer 1 receptive field plot for $\beta = .43$.

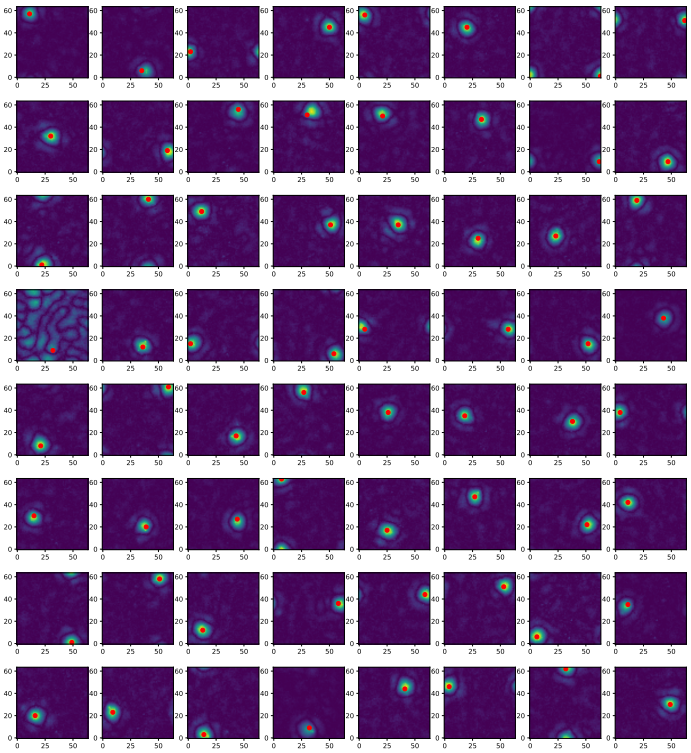


Figure A.9: Layer 3 receptive field plot for $\beta = .395$.

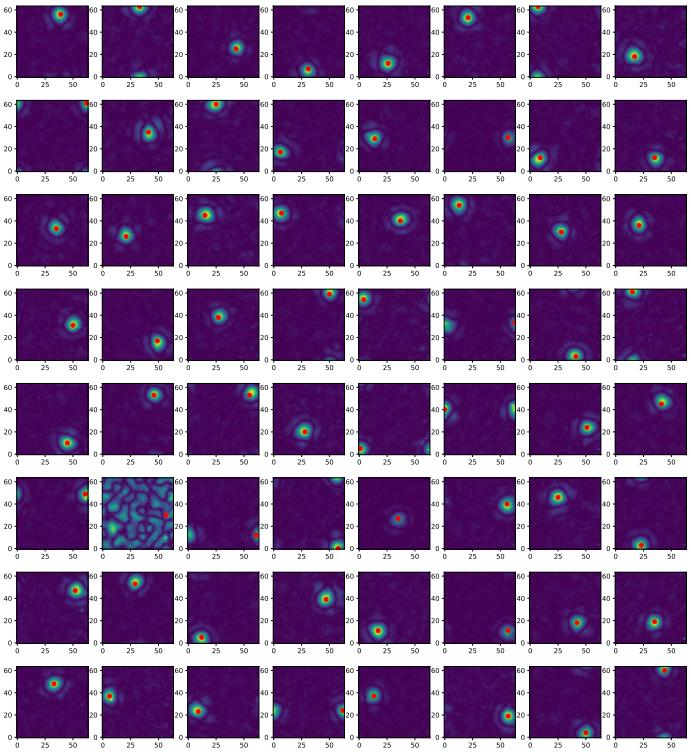


Figure A.10: Layer 3 receptive field plot for $\beta = .4$.

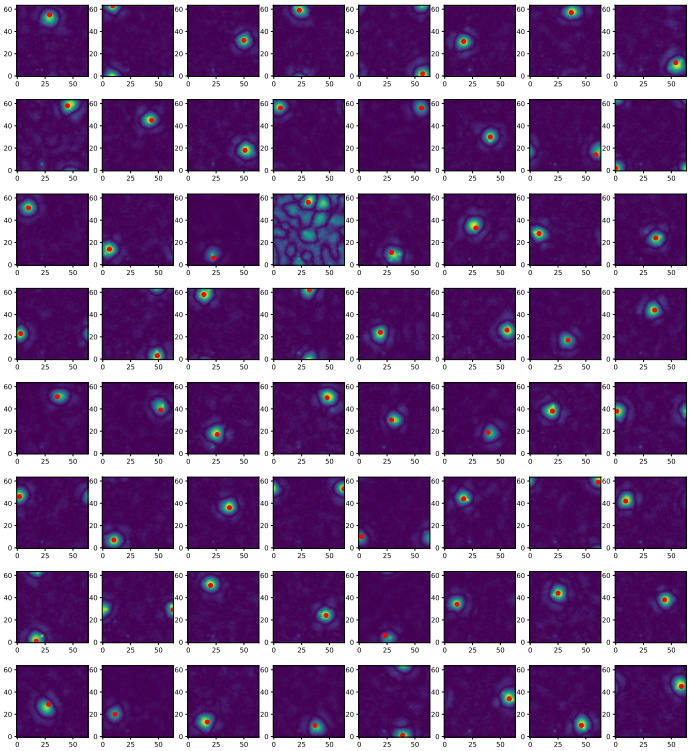


Figure A.11: Layer 3 receptive field plot for $\beta = .405$.

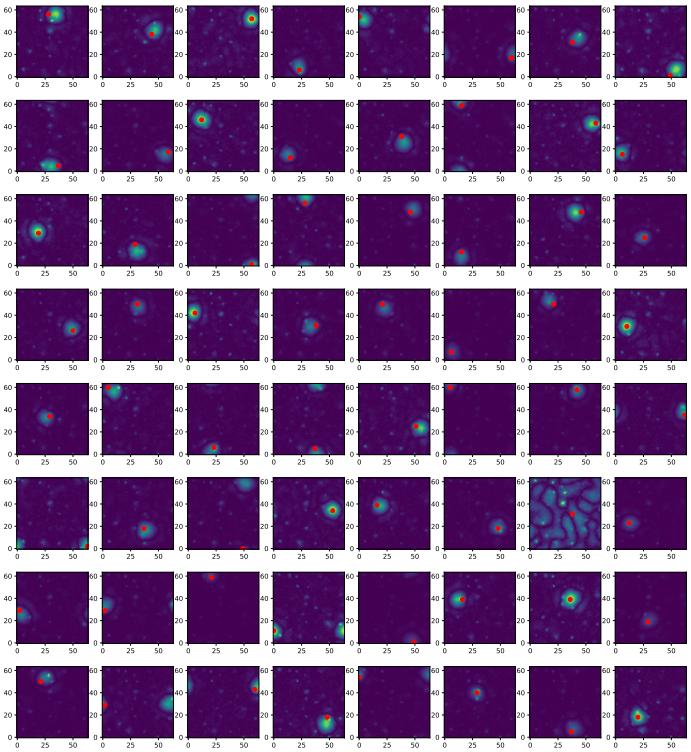


Figure A.12: Layer 3 receptive field plot for $\beta = .41$.

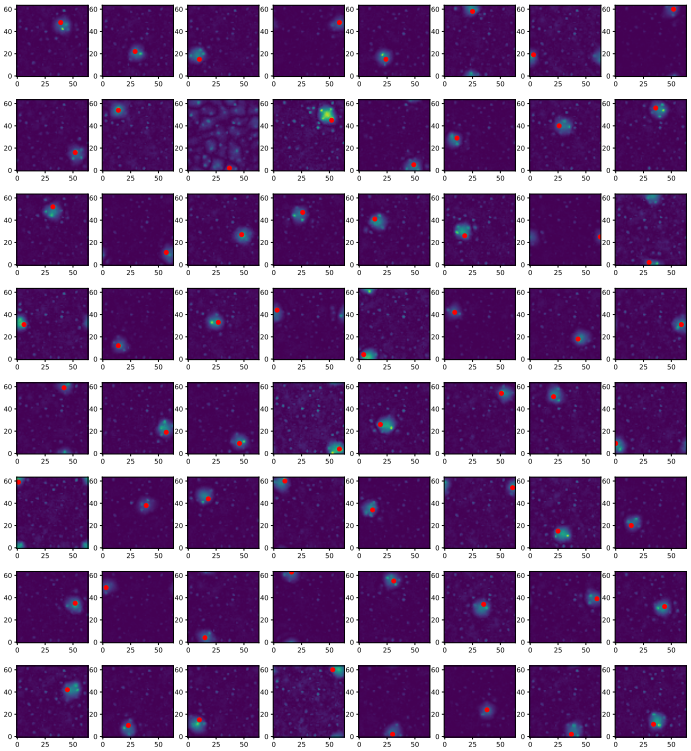


Figure A.13: Layer 3 receptive field plot for $\beta = .415$.

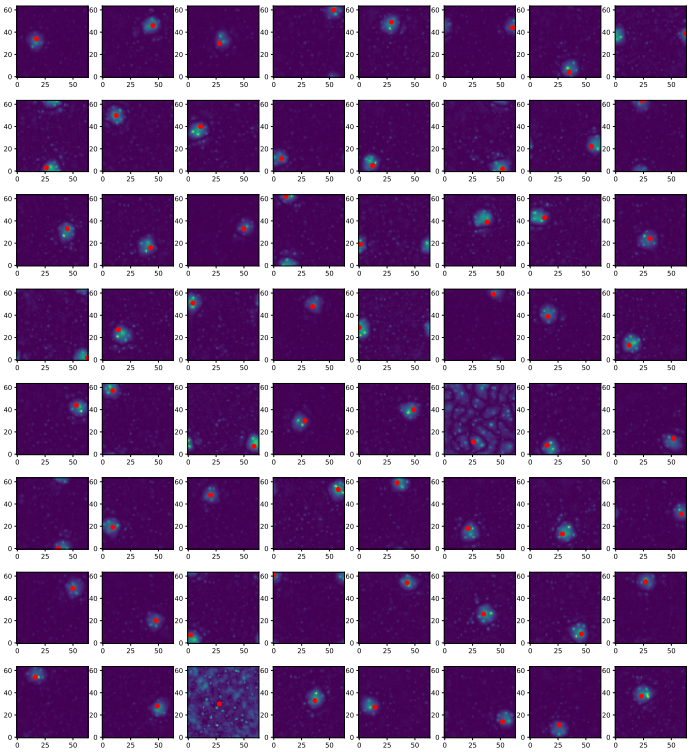


Figure A.14: Layer 3 receptive field plot for $\beta = .42$.

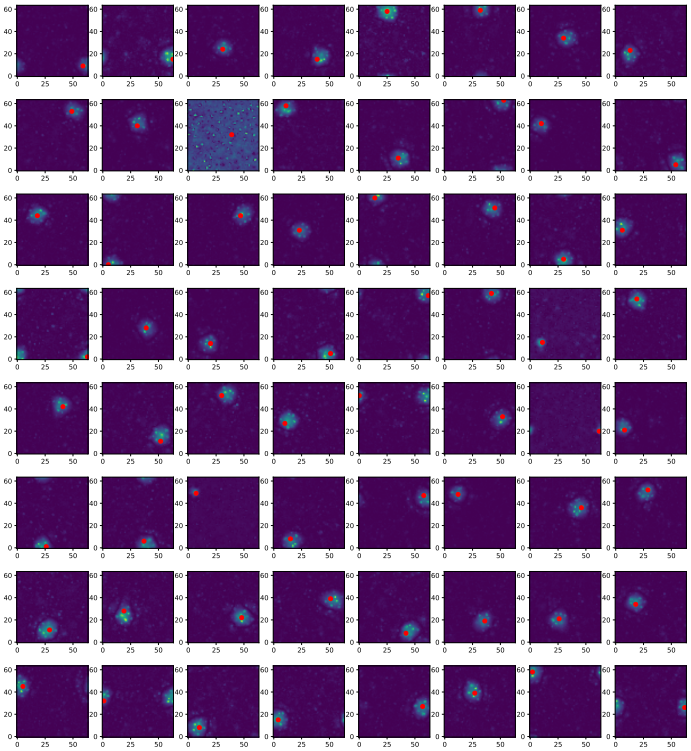


Figure A.15: Layer 3 receptive field plot for $\beta = .425$.

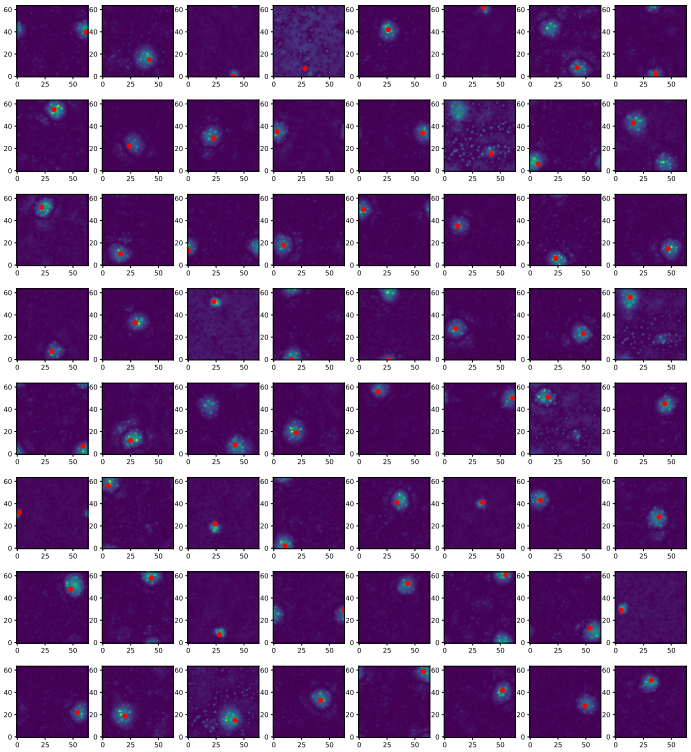
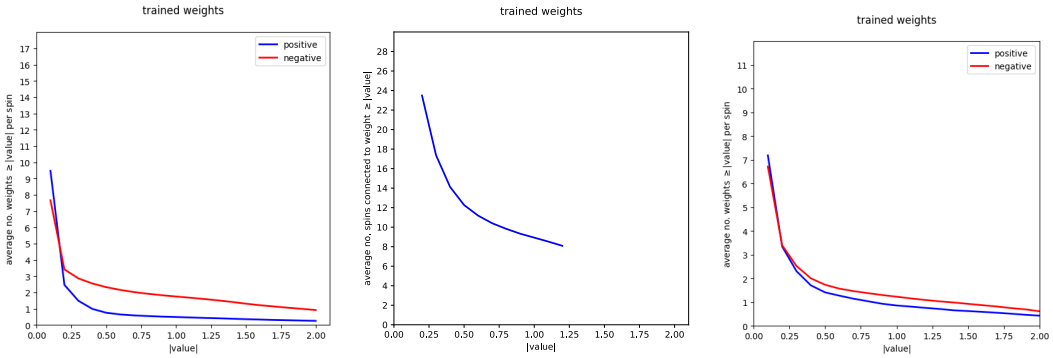


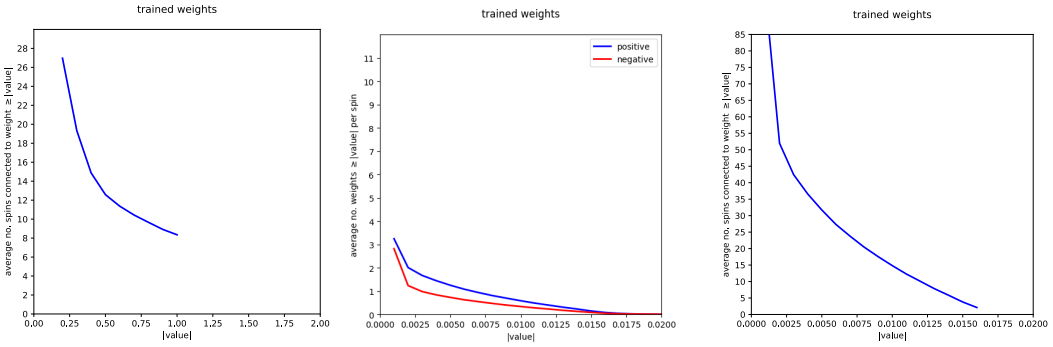
Figure A.16: Layer 3 receptive field plot for $\beta = .43$.

A.2 Weight Analysis Plots

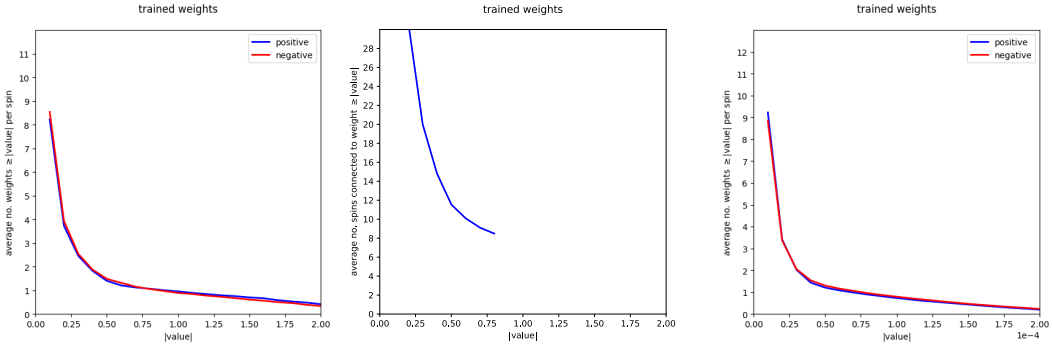
The plots starting on the next page analyze the weight structure for each value of β, analyzing both the weight tensors and receptive field tensors. Each tensor has two plots. The first is the average number of trained positive/negative weights with a magnitude above a given value connected to a given spin location in an input lattice. The second is the average number of spin locations connected to each weight with magnitudes above a given value, such that all locations connect to at least two weights. More information about this analysis and its results are in Section 4.3.



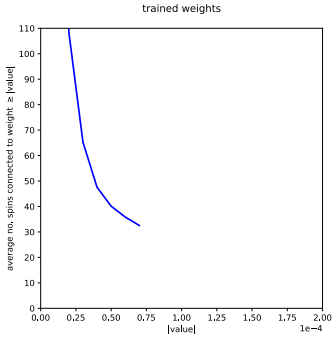
(a) Layer 1 Weights (b) Layer 1 Weights (c) Layer 2 Weights



(d) Layer 2 Weights (e) Layer 2 Receptive (f) Layer 2 Receptive

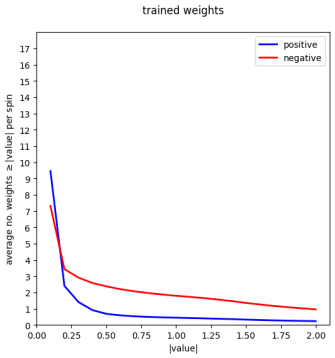


(g) Layer 3 Weights (h) Layer 3 Weights (i) Layer 3 Receptive

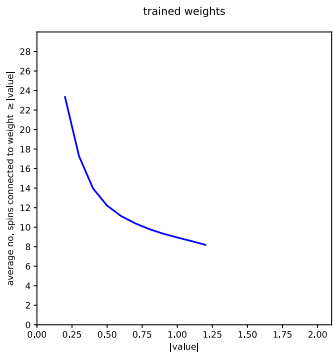


(j) Layer 3 Receptive

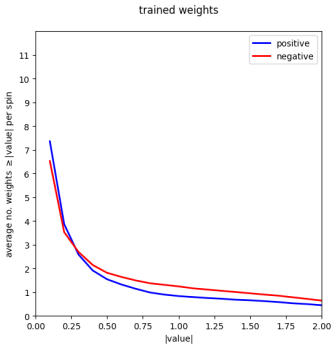
Figure A.17: Weight analysis plot for $\beta = .395$.



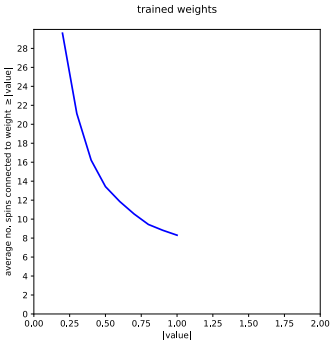
(a) Layer 1 Weights



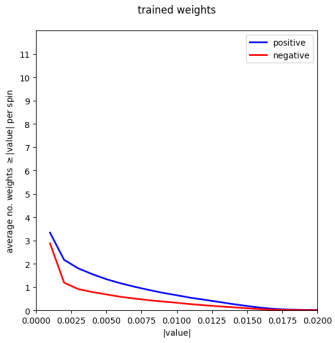
(b) Layer 1 Weights



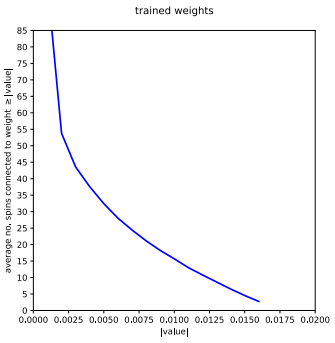
(c) Layer 2 Weights



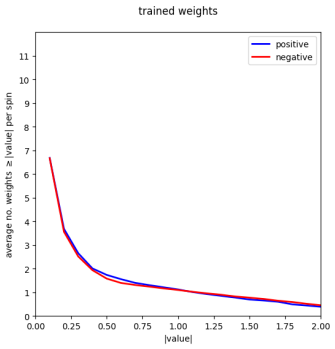
(d) Layer 2 Weights



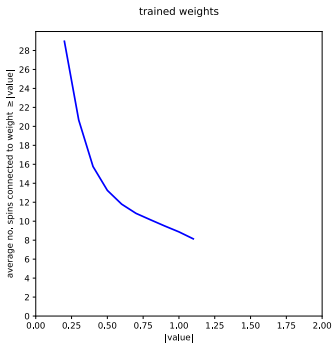
(e) Layer 2 Receptive



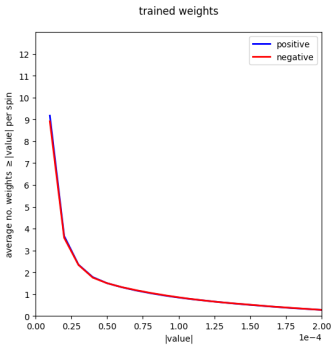
(f) Layer 2 Receptive



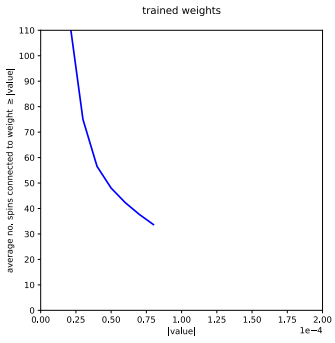
(g) Layer 3 Weights



(h) Layer 3 Weights

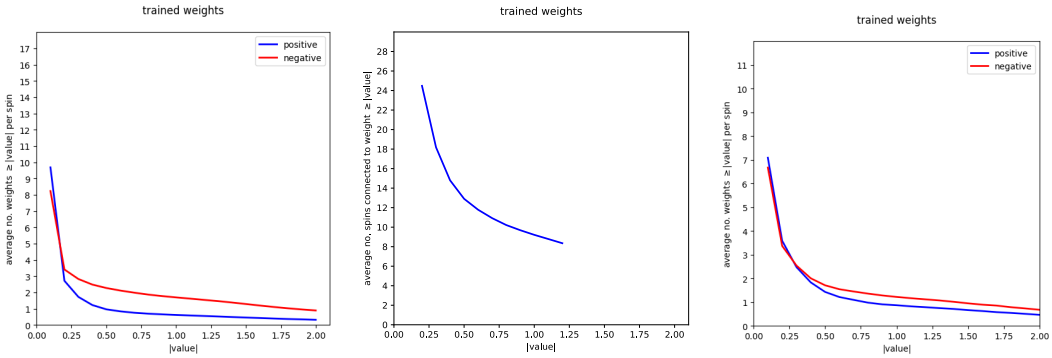


(i) Layer 3 Receptive

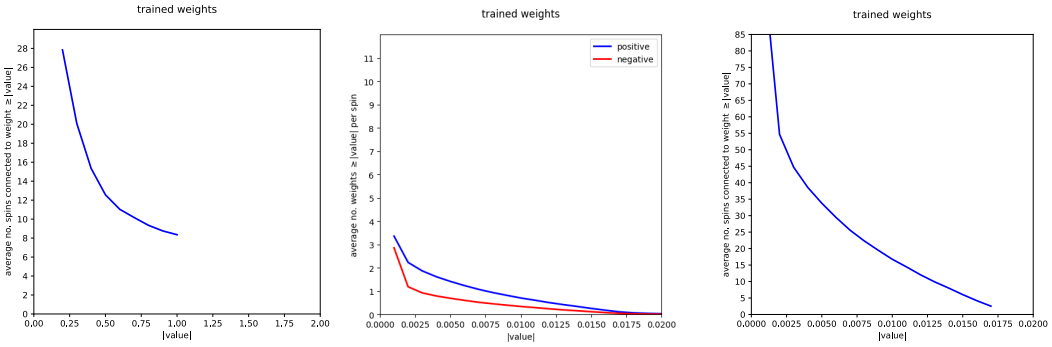


(j) Layer 3 Receptive

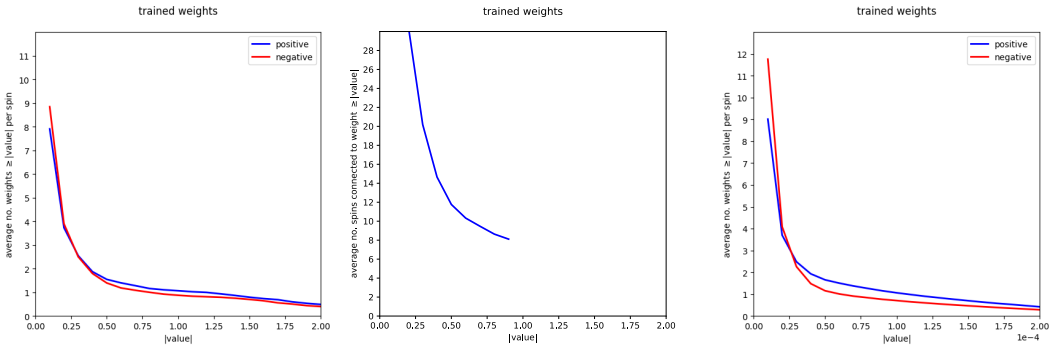
Figure A.18: Weight analysis plot for $\beta = .4$.



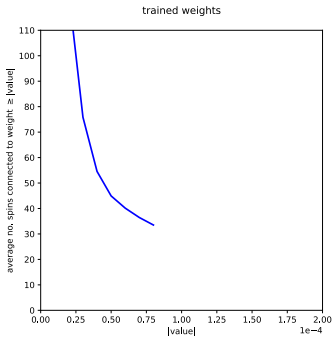
(a) Layer 1 Weights (b) Layer 1 Weights (c) Layer 2 Weights



(d) Layer 2 Weights (e) Layer 2 Receptive (f) Layer 2 Receptive

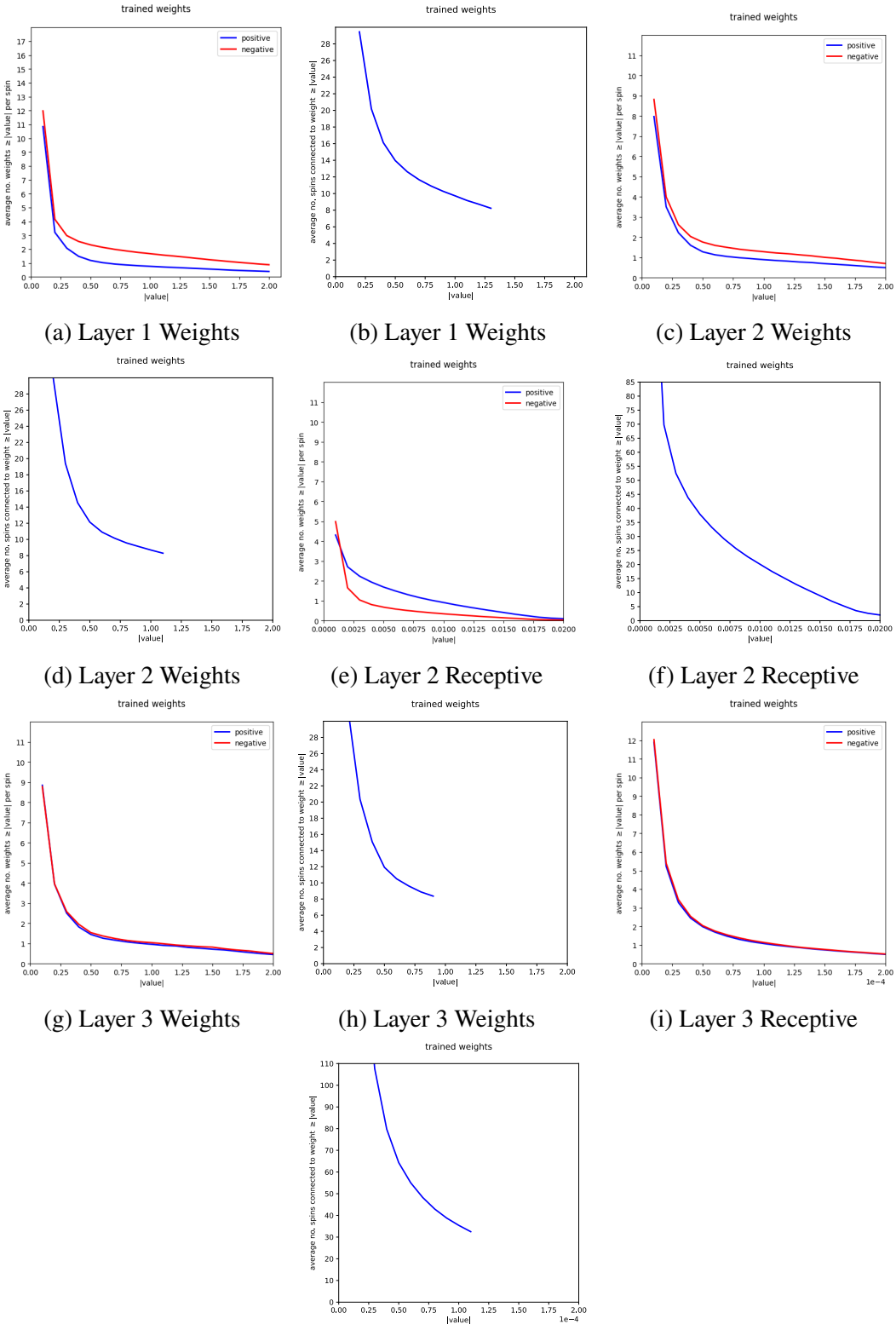


(g) Layer 3 Weights (h) Layer 3 Weights (i) Layer 3 Receptive



(j) Layer 3 Receptive

Figure A.19: Weight analysis plot for $\beta = .405$.



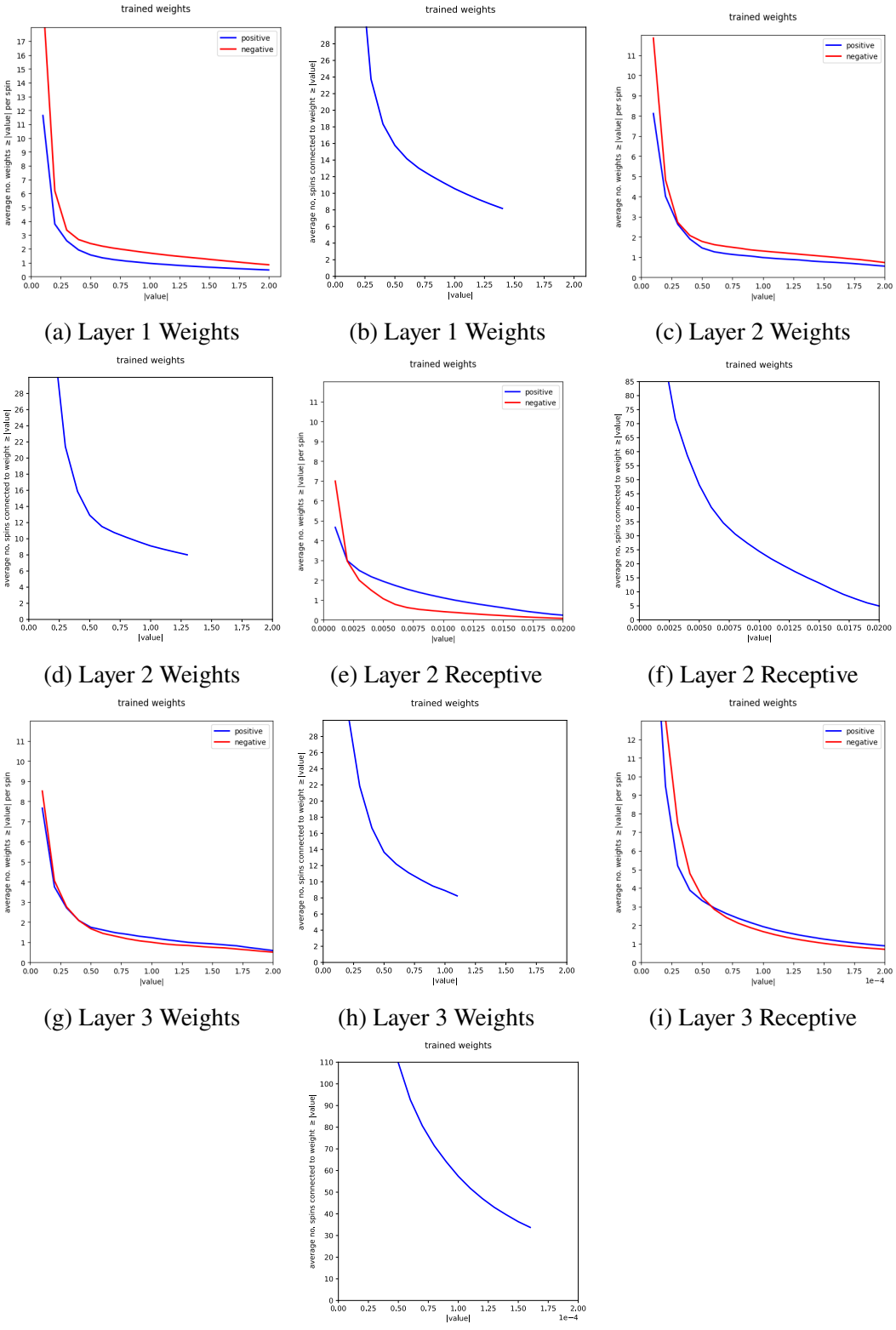
(a) Layer 1 Weights (b) Layer 1 Weights (c) Layer 2 Weights

(d) Layer 2 Weights (e) Layer 2 Receptive (f) Layer 2 Receptive

(g) Layer 3 Weights (h) Layer 3 Weights (i) Layer 3 Receptive

(j) Layer 3 Receptive

Figure A.20: Weight analysis plot for $\beta = .41$.



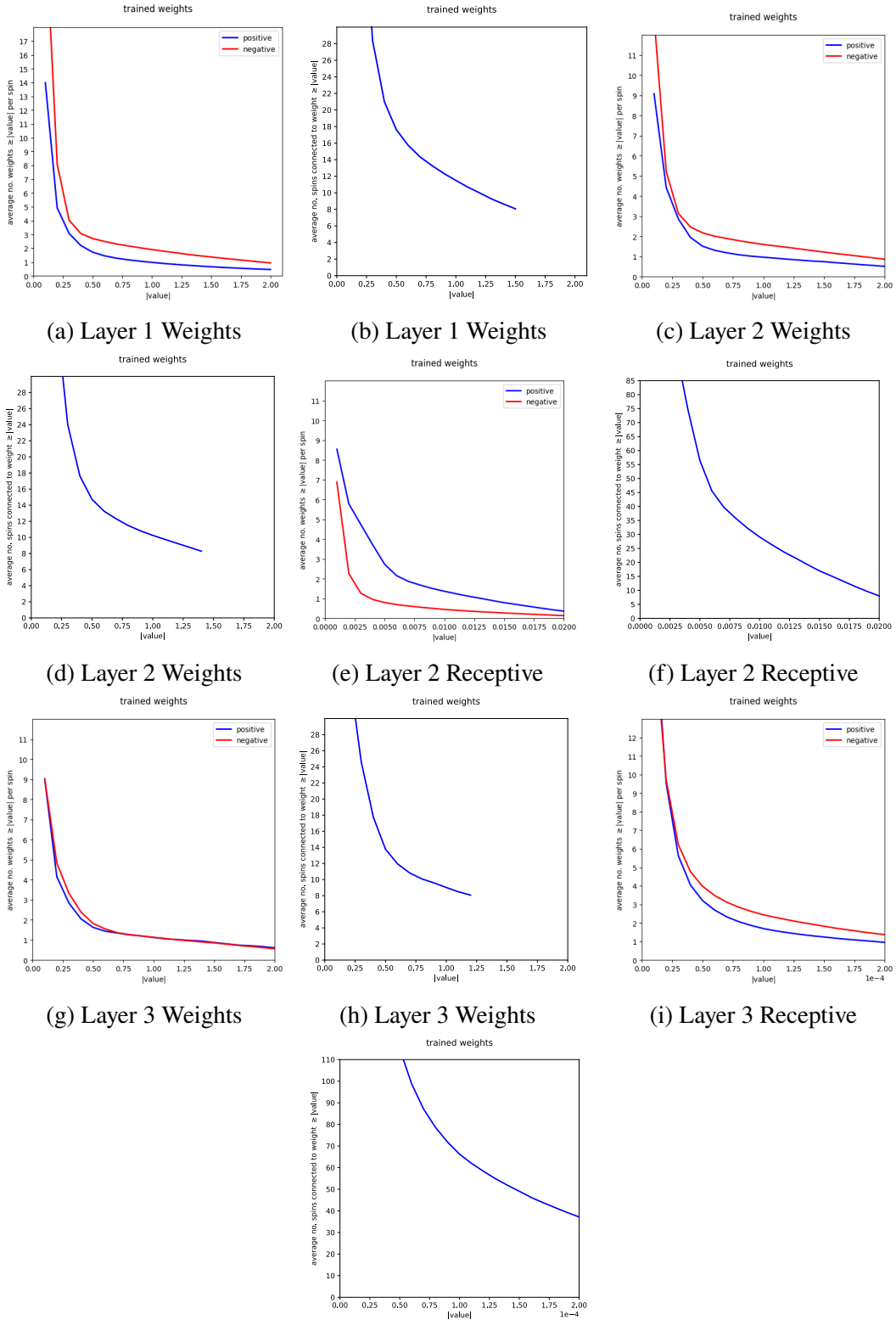
(a) Layer 1 Weights (b) Layer 1 Weights (c) Layer 2 Weights

(d) Layer 2 Weights (e) Layer 2 Receptive (f) Layer 2 Receptive

(g) Layer 3 Weights (h) Layer 3 Weights (i) Layer 3 Receptive

(j) Layer 3 Receptive

Figure A.21: Weight analysis plot for $\beta = .415$.



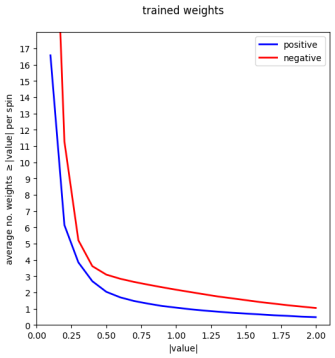
(a) Layer 1 Weights (b) Layer 1 Weights (c) Layer 2 Weights

(d) Layer 2 Weights (e) Layer 2 Receptive (f) Layer 2 Receptive

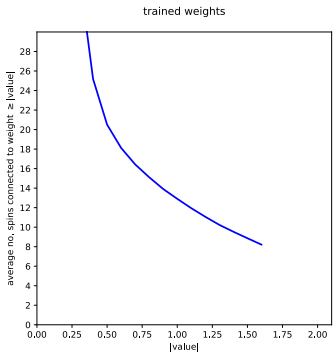
(g) Layer 3 Weights (h) Layer 3 Weights (i) Layer 3 Receptive

(j) Layer 3 Receptive

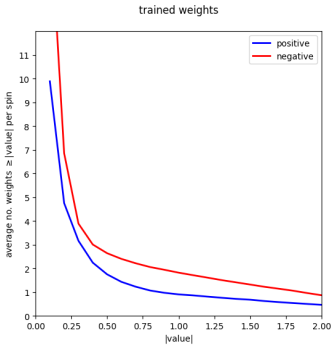
Figure A.22: Weight analysis plot for $\beta = .42$.



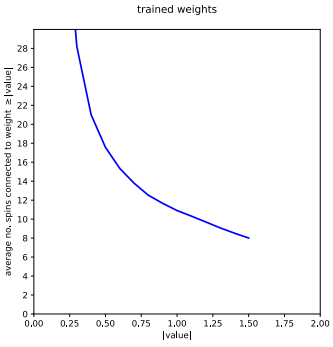
(a) Layer 1 Weights



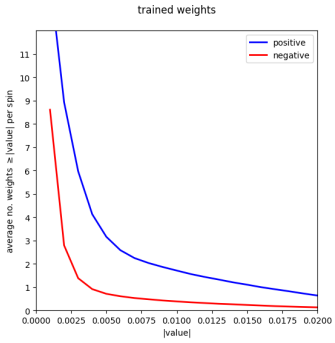
(b) Layer 1 Weights



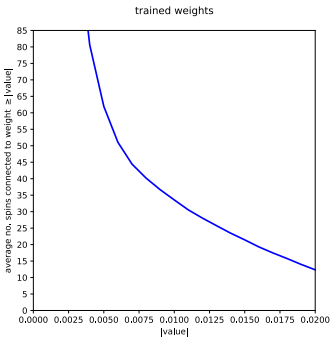
(c) Layer 2 Weights



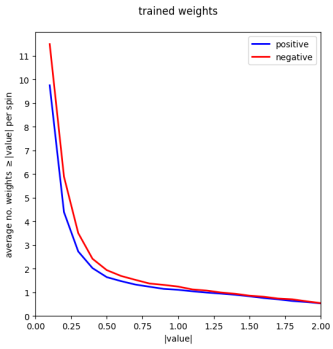
(d) Layer 2 Weights



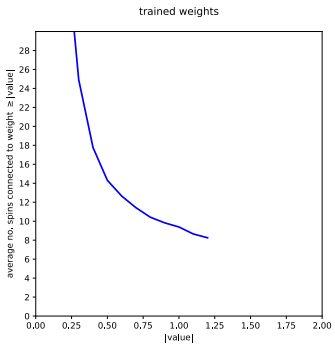
(e) Layer 2 Receptive



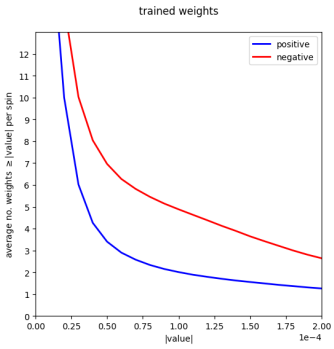
(f) Layer 2 Receptive



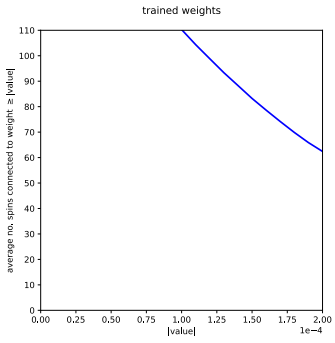
(g) Layer 3 Weights



(h) Layer 3 Weights

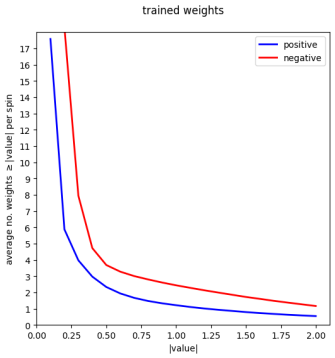


(i) Layer 3 Receptive

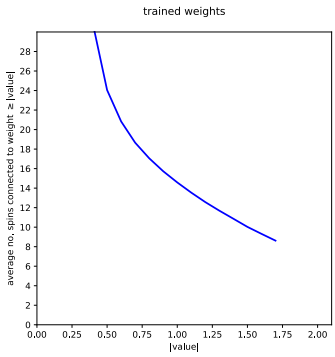


(j) Layer 3 Receptive

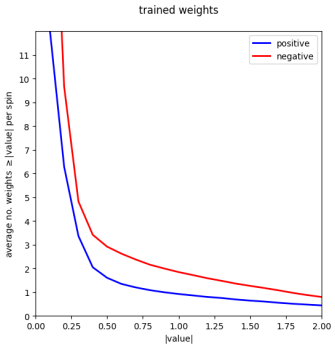
Figure A.23: Weight analysis plot for $\beta = .425$.



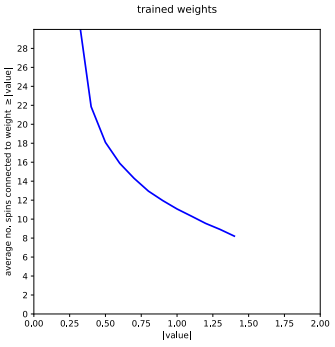
(a) Layer 1 Weights



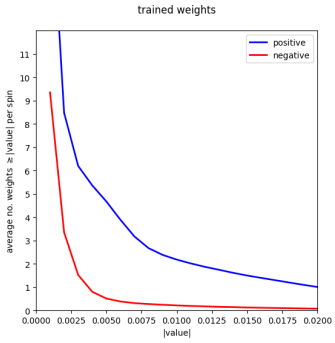
(b) Layer 1 Weights



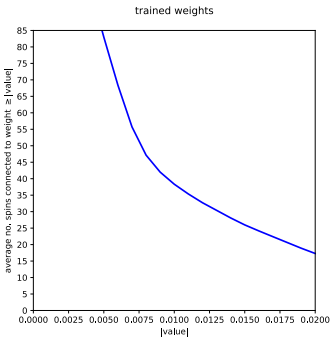
(c) Layer 2 Weights



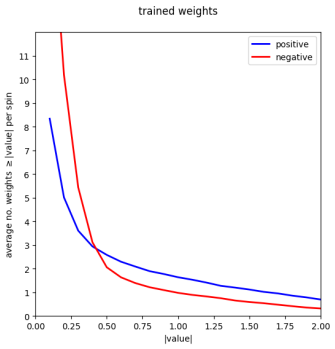
(d) Layer 2 Weights



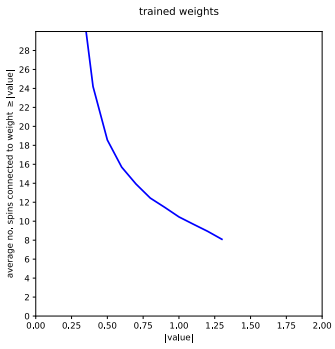
(e) Layer 2 Receptive



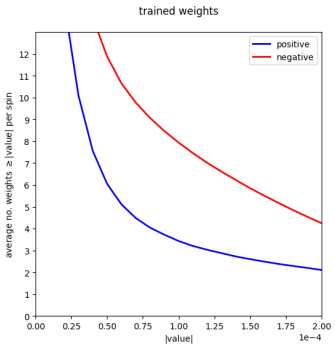
(f) Layer 2 Receptive



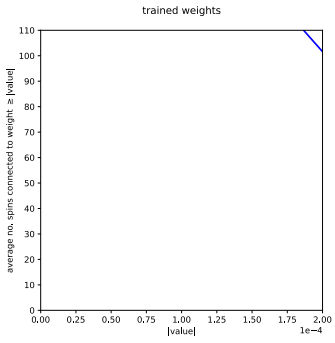
(g) Layer 3 Weights



(h) Layer 3 Weights



(i) Layer 3 Receptive

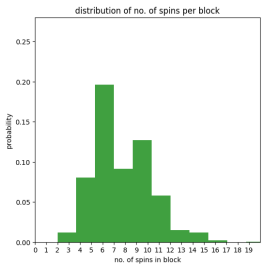


(j) Layer 3 Receptive

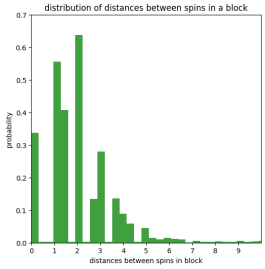
Figure A.24: Weight analysis plot for $\beta = .43$.

A.3 Block Analysis Histograms

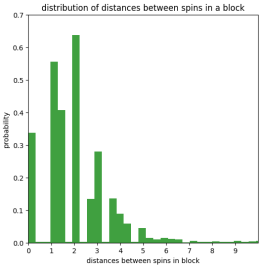
The plots starting on the next page analyze the blocking structure for each value of β, analyzing the structure for both the weight tensors and receptive field tensors. Each tensor has three histograms. The first histogram is the number of spins per block averaged over all the lattices. The second is the maximum distance between spins per block averaged over all the lattices. The third is just all the distances between spins averaged over all the lattices. These plots are discussed in detail in Section 4.4.



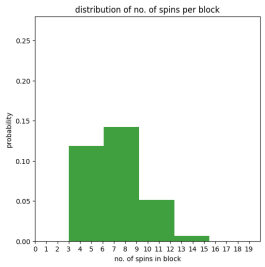
(a) Layer 1



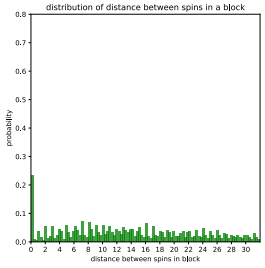
(b) Layer 1



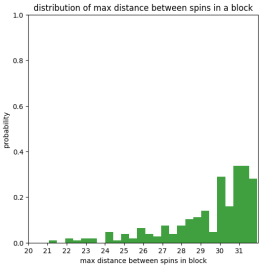
(c) Layer 1



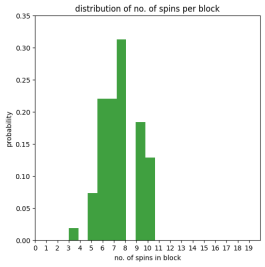
(d) Layer 2 Weights



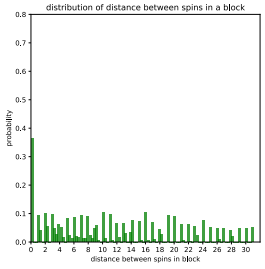
(e) Layer 2 Weights



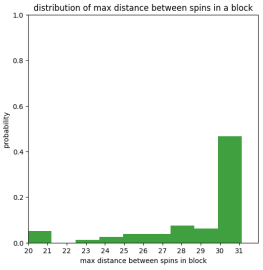
(f) Layer 2 Weights



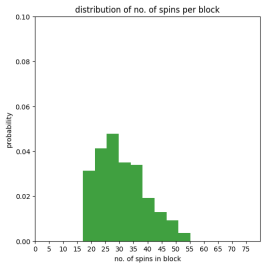
(g) Layer 3 Weights



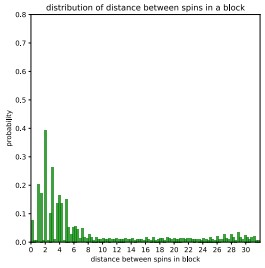
(h) Layer 3 Weights



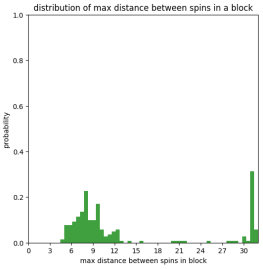
(i) Layer 3 Weights



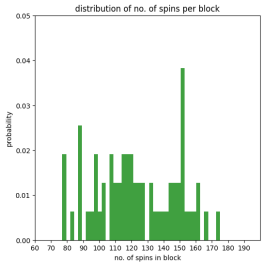
(j) Layer 2 Receptive



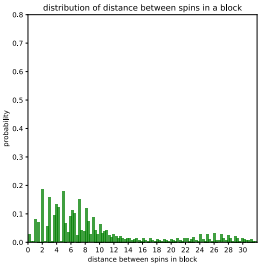
(k) Layer 2 Receptive



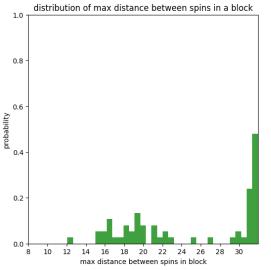
(l) Layer 2 Receptive



(m) Layer 3 Receptive

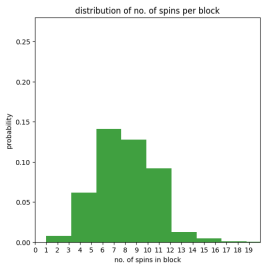


(n) Layer 3 Receptive

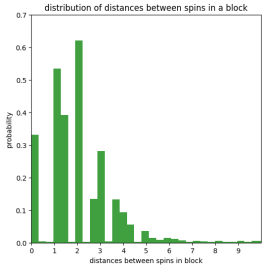


(o) Layer 3 Receptive

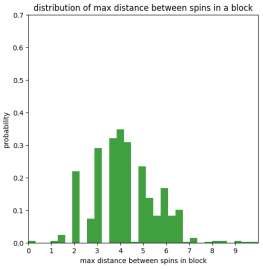
Figure A.25: Block analysis plot for $\beta = .395$.



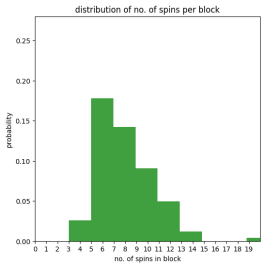
(a) Layer 1



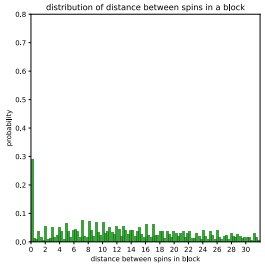
(b) Layer 1



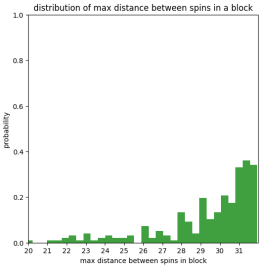
(c) Layer 1



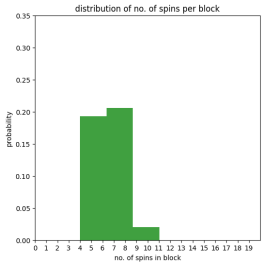
(d) Layer 2 Weights



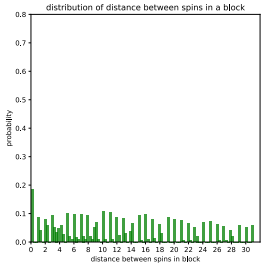
(e) Layer 2 Weights



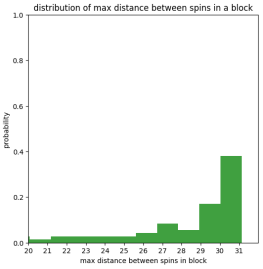
(f) Layer 2 Weights



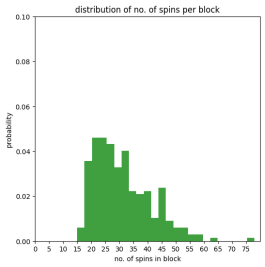
(g) Layer 3 Weights



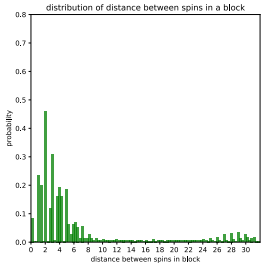
(h) Layer 3 Weights



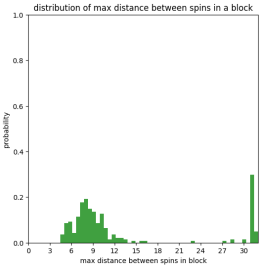
(i) Layer 3 Weights



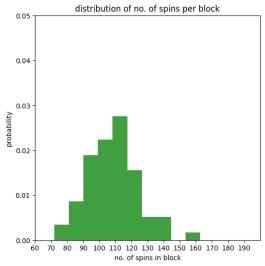
(j) Layer 2 Receptive



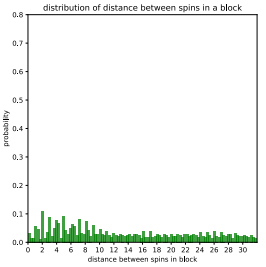
(k) Layer 2 Receptive



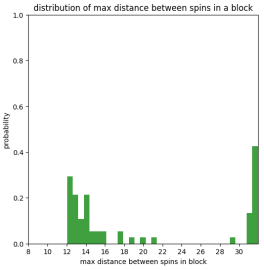
(l) Layer 2 Receptive



(m) Layer 3 Receptive

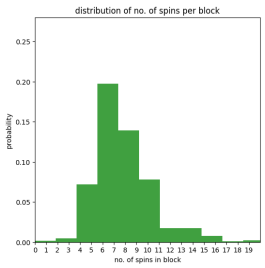


(n) Layer 3 Receptive

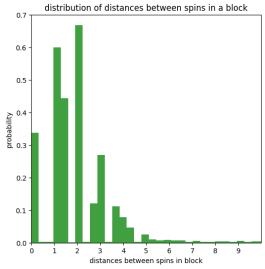


(o) Layer 3 Receptive

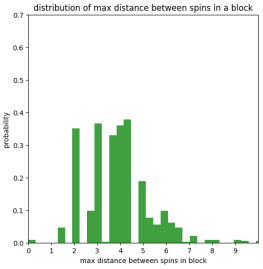
Figure A.26: Block analysis plot for $\beta = .4$.



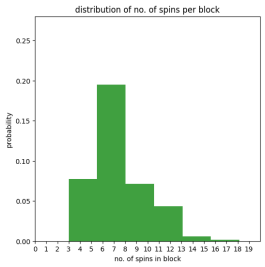
(a) Layer 1



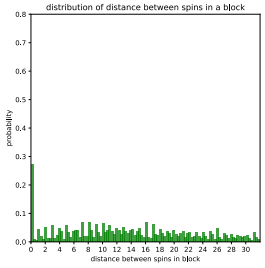
(b) Layer 1



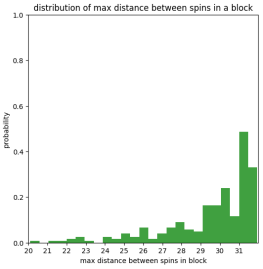
(c) Layer 1



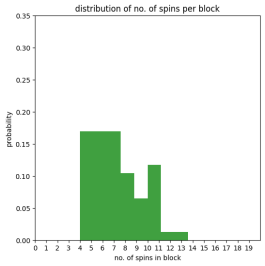
(d) Layer 2 Weights



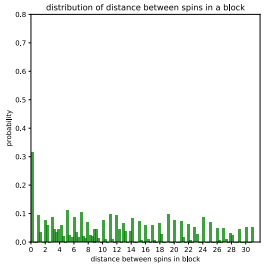
(e) Layer 2 Weights



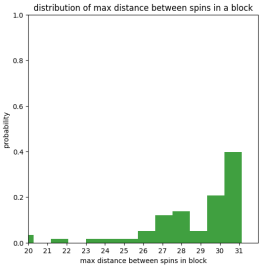
(f) Layer 2 Weights



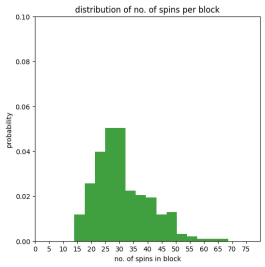
(g) Layer 3 Weights



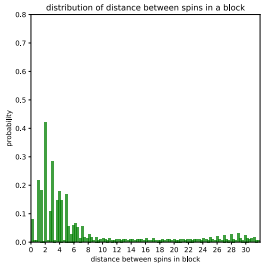
(h) Layer 3 Weights



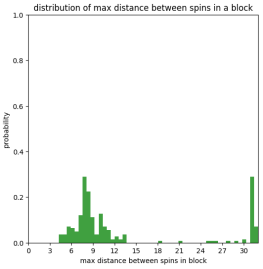
(i) Layer 3 Weights



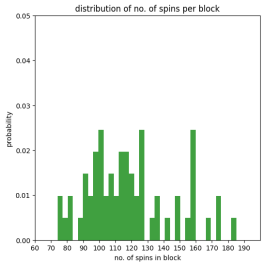
(j) Layer 2 Receptive



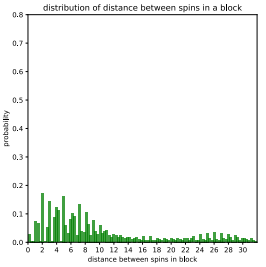
(k) Layer 2 Receptive



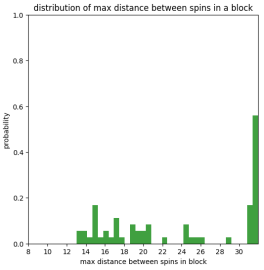
(l) Layer 2 Receptive



(m) Layer 3 Receptive

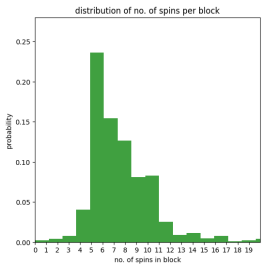


(n) Layer 3 Receptive

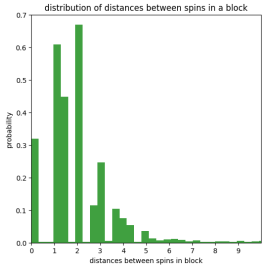


(o) Layer 3 Receptive

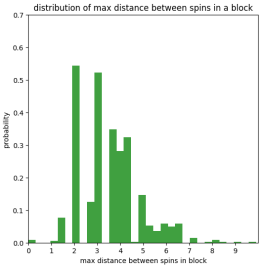
Figure A.27: Block analysis plot for $\beta = .405$.



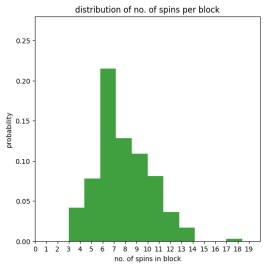
(a) Layer 1



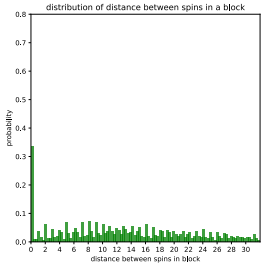
(b) Layer 1



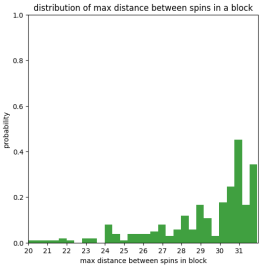
(c) Layer 1



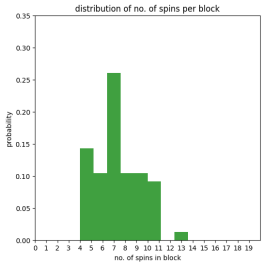
(d) Layer 2 Weights



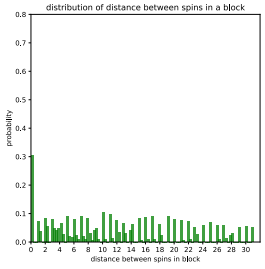
(e) Layer 2 Weights



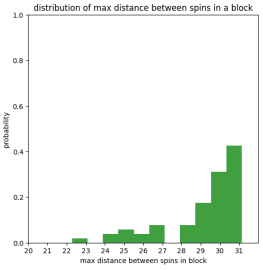
(f) Layer 2 Weights



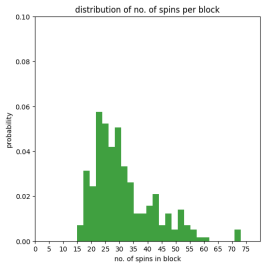
(g) Layer 3 Weights



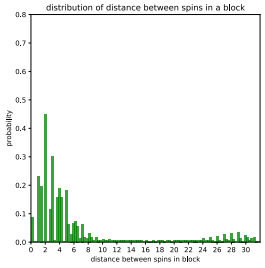
(h) Layer 3 Weights



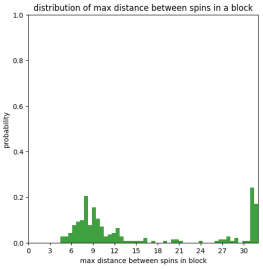
(i) Layer 3 Weights



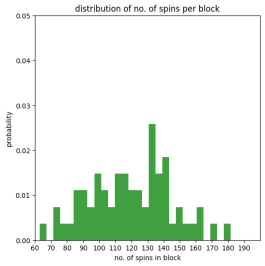
(j) Layer 2 Receptive



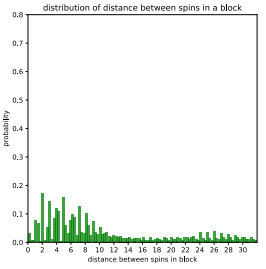
(k) Layer 2 Receptive



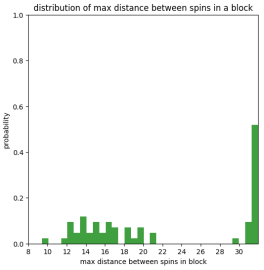
(l) Layer 2 Receptive



(m) Layer 3 Receptive

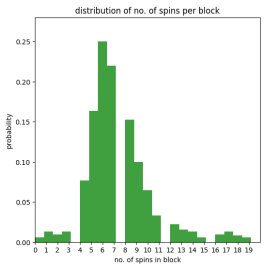


(n) Layer 3 Receptive

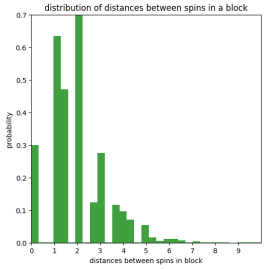


(o) Layer 3 Receptive

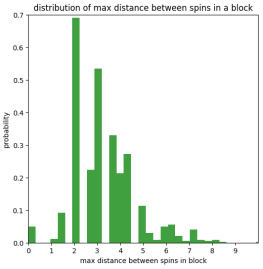
Figure A.28: Block analysis plot for $\beta = .41$.



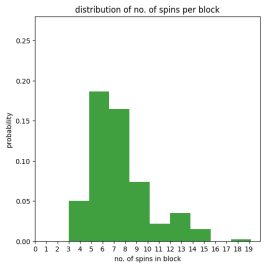
(a) Layer 1



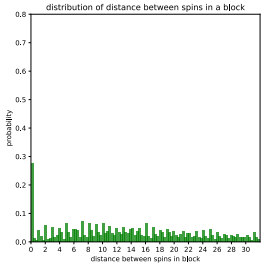
(b) Layer 1



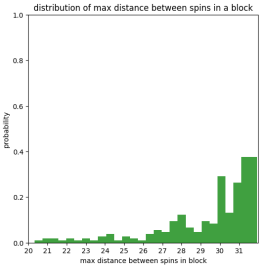
(c) Layer 1



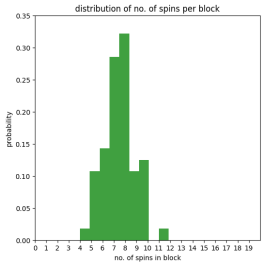
(d) Layer 2 Weights



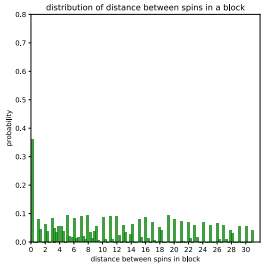
(e) Layer 2 Weights



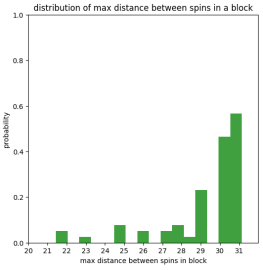
(f) Layer 2 Weights



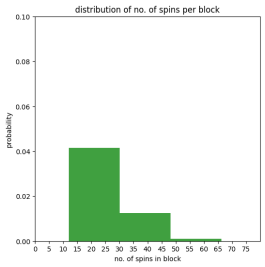
(g) Layer 3 Weights



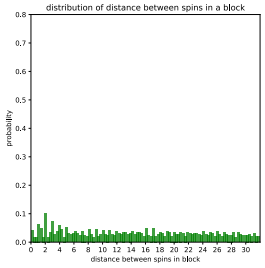
(h) Layer 3 Weights



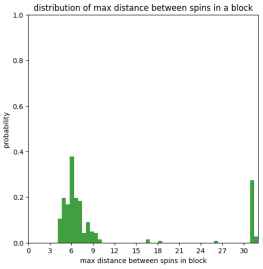
(i) Layer 3 Weights



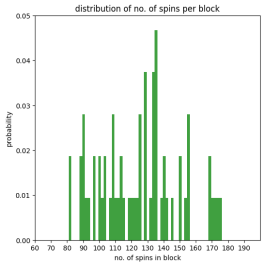
(j) Layer 2 Receptive



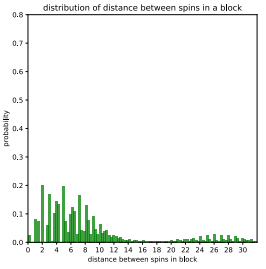
(k) Layer 2 Receptive



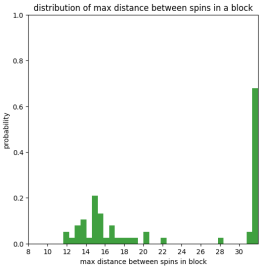
(l) Layer 2 Receptive



(m) Layer 3 Receptive

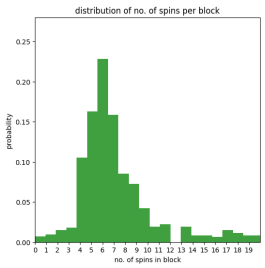


(n) Layer 3 Receptive

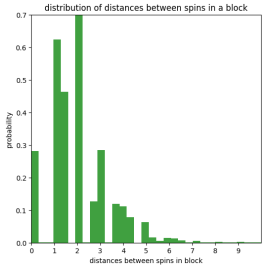


(o) Layer 3 Receptive

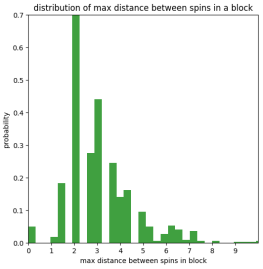
Figure A.29: Block analysis plot for $\beta = .415$.



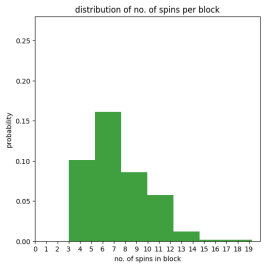
(a) Layer 1



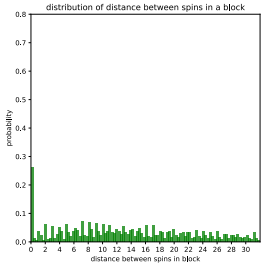
(b) Layer 1



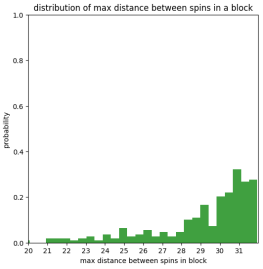
(c) Layer 1



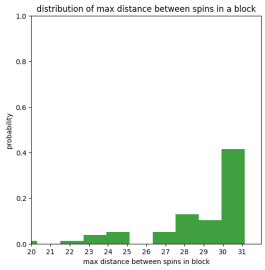
(d) Layer 2 Weights



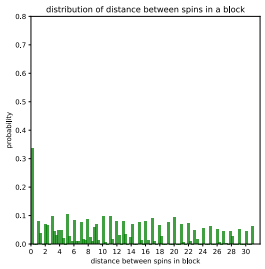
(e) Layer 2 Weights



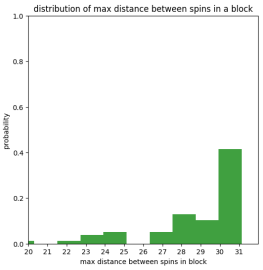
(f) Layer 2 Weights



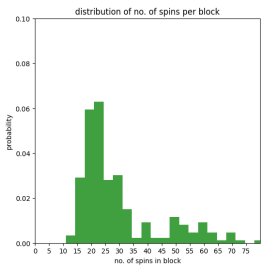
(g) Layer 3 Weights



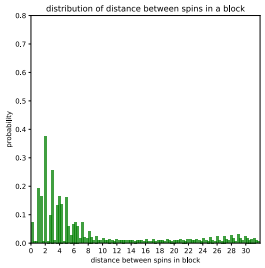
(h) Layer 3 Weights



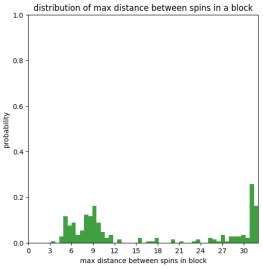
(i) Layer 3 Weights



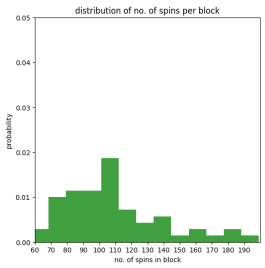
(j) Layer 2 Receptive



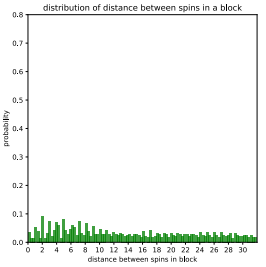
(k) Layer 2 Receptive



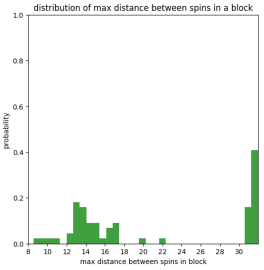
(l) Layer 2 Receptive



(m) Layer 3 Receptive

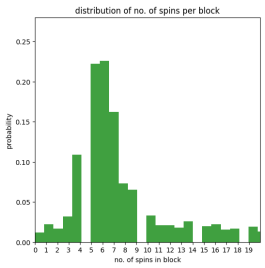


(n) Layer 3 Receptive

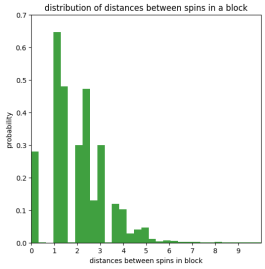


(o) Layer 3 Receptive

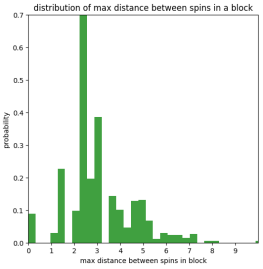
Figure A.30: Block analysis plot for $\beta = .42$.



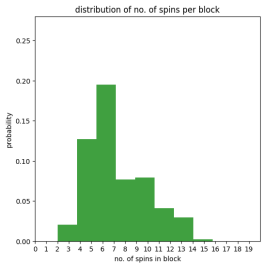
(a) Layer 1



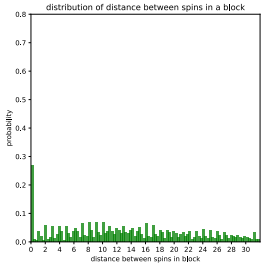
(b) Layer 1



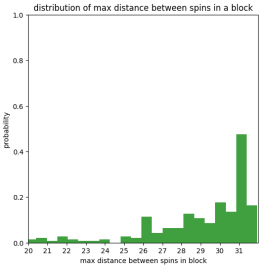
(c) Layer 1



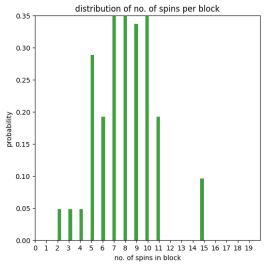
(d) Layer 2 Weights



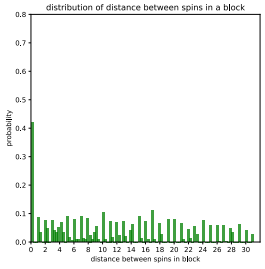
(e) Layer 2 Weights



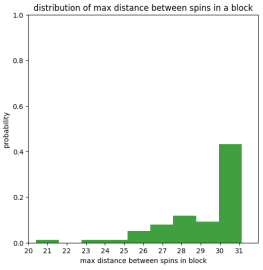
(f) Layer 2 Weights



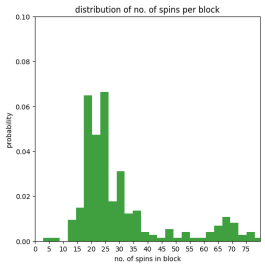
(g) Layer 3 Weights



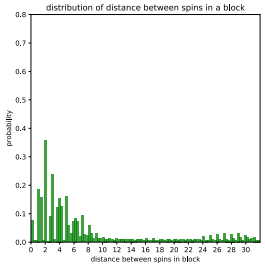
(h) Layer 3 Weights



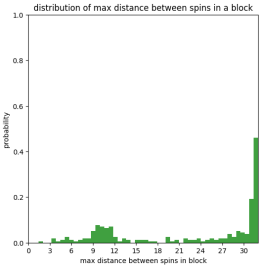
(i) Layer 3 Weights



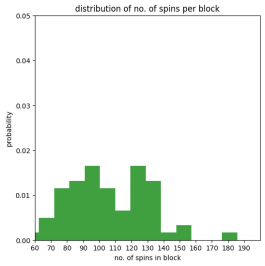
(j) Layer 2 Receptive



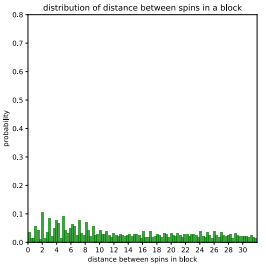
(k) Layer 2 Receptive



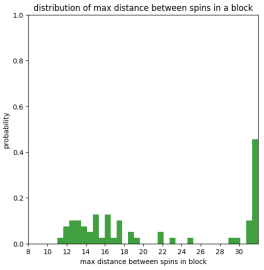
(l) Layer 2 Receptive



(m) Layer 3 Receptive

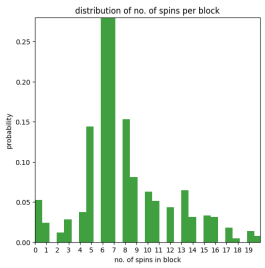


(n) Layer 3 Receptive

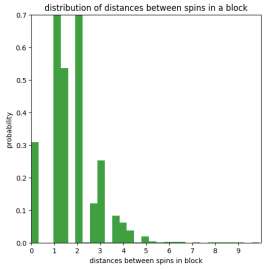


(o) Layer 3 Receptive

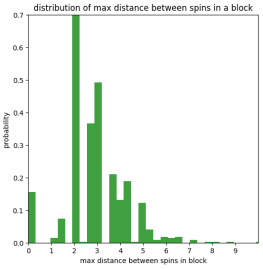
Figure A.31: Block analysis plot for $\beta = .425$.



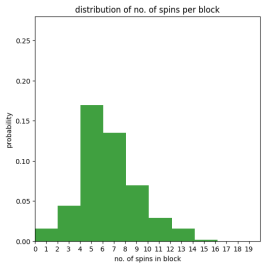
(a) Layer 1



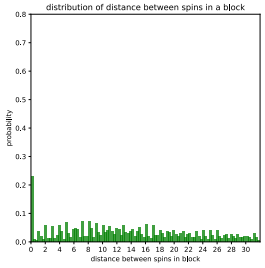
(b) Layer 1



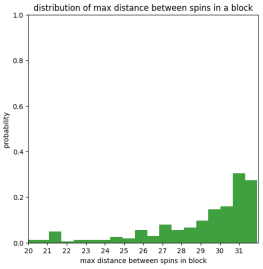
(c) Layer 1



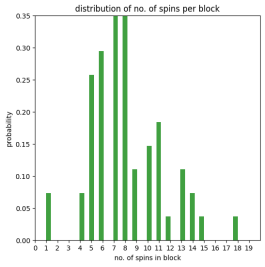
(d) Layer 2 Weights



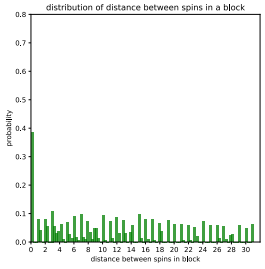
(e) Layer 2 Weights



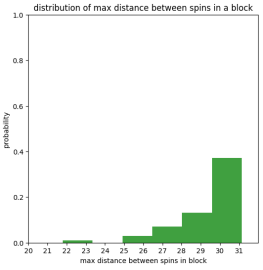
(f) Layer 2 Weights



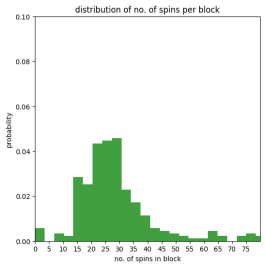
(g) Layer 3 Weights



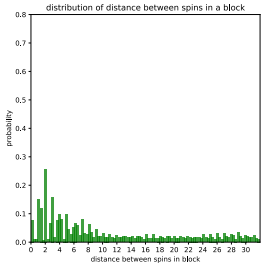
(h) Layer 3 Weights



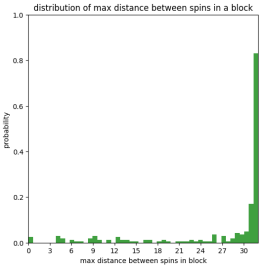
(i) Layer 3 Receptive



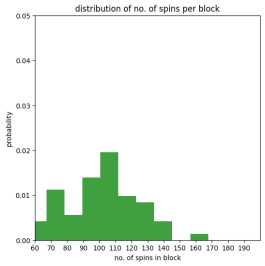
(j) Layer 2 Receptive



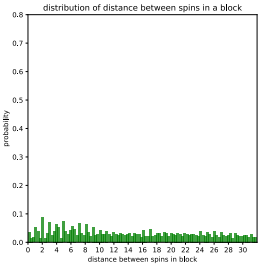
(k) Layer 2 Receptive



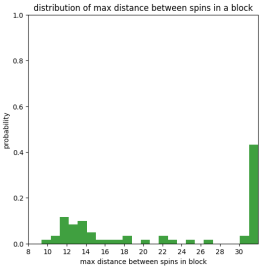
(l) Layer 2 Receptive



(m) Layer 3 Receptive



(n) Layer 3 Receptive



(o) Layer 3 Receptive

Figure A.32: Block analysis plot for $\beta = .43$.

A.4 Reconstruction Plots

This section contains plots made by using the deep learning network as an autoencoder to reconstruct the original Ising model. For various values of β In each plot, the first row consists of the original Ising input, rows 2-4 consist of the reconstructed Ising models using 1, 2, and 3 layers respectively, and rows 5-7 do the same thing but only using the largest two receptive field values to calculate the reconstruction. These results are discussed in detail in Section 4.5.

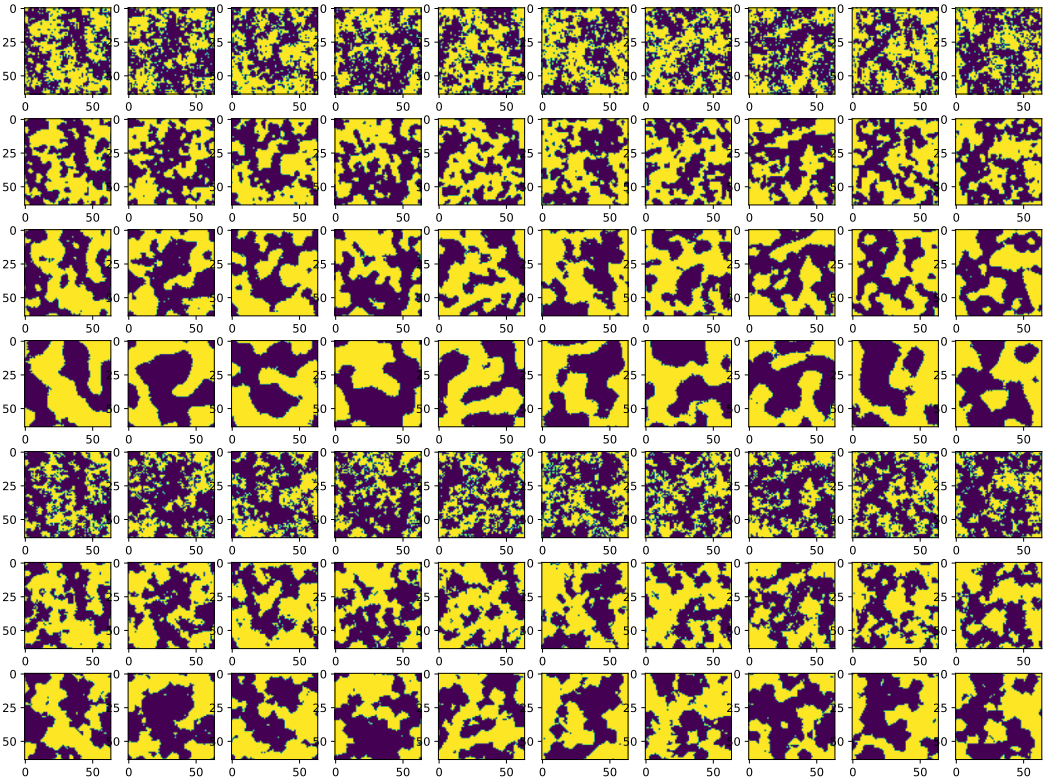


Figure A.33: Reconstructed model for $\beta = .395$.

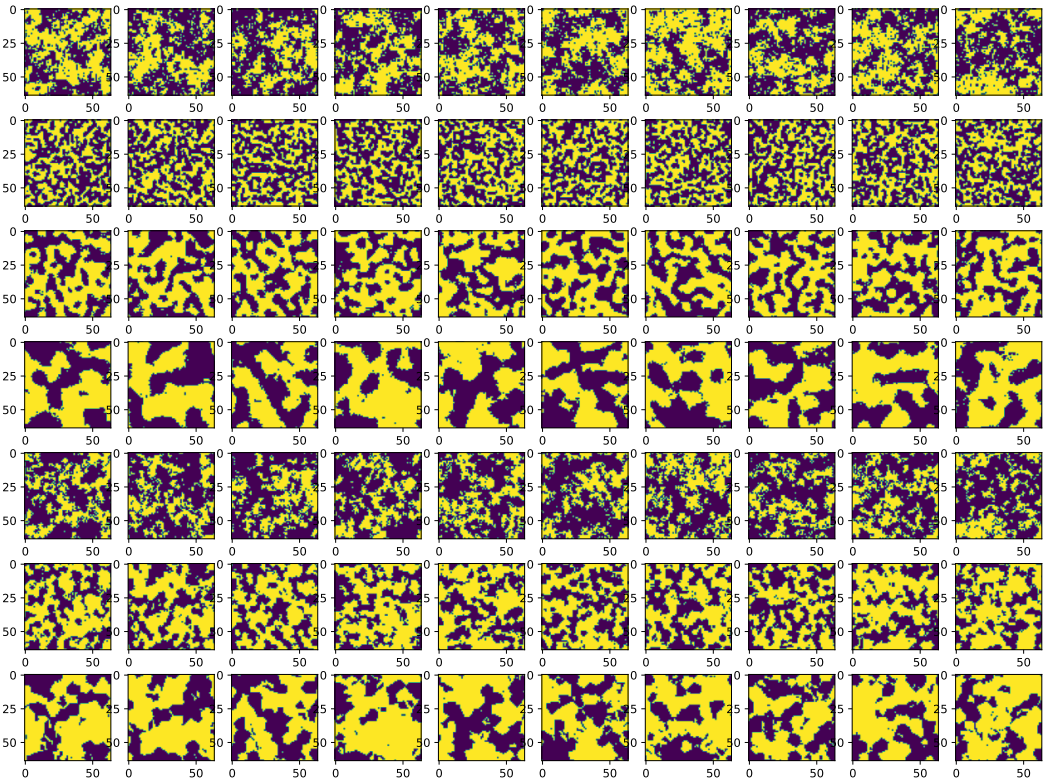


Figure A.34: Reconstructed model for $\beta = .4$.

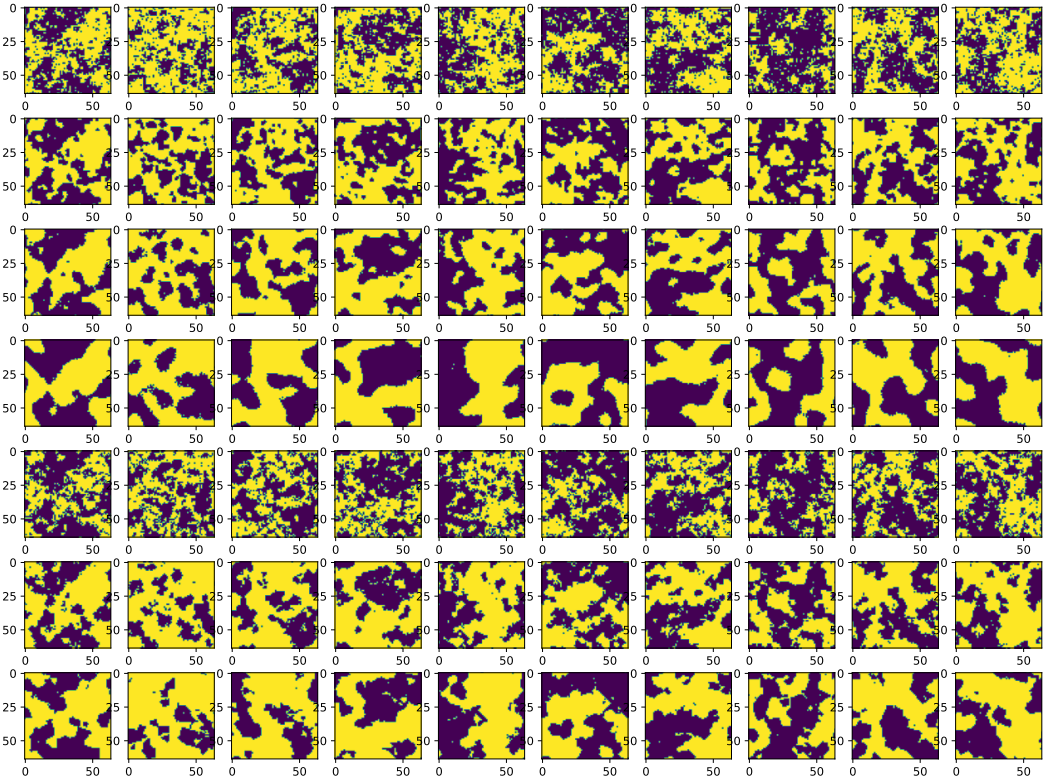


Figure A.35: Reconstructed model for $\beta = .405$.

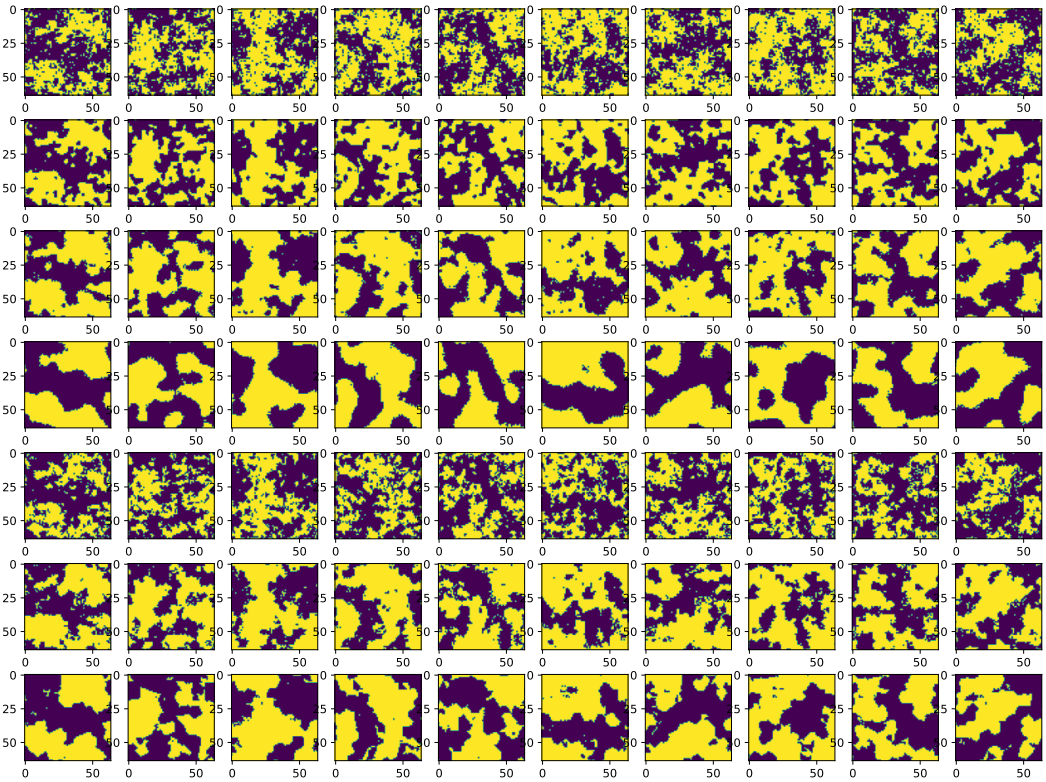


Figure A.36: Reconstructed model for $\beta = .41$.

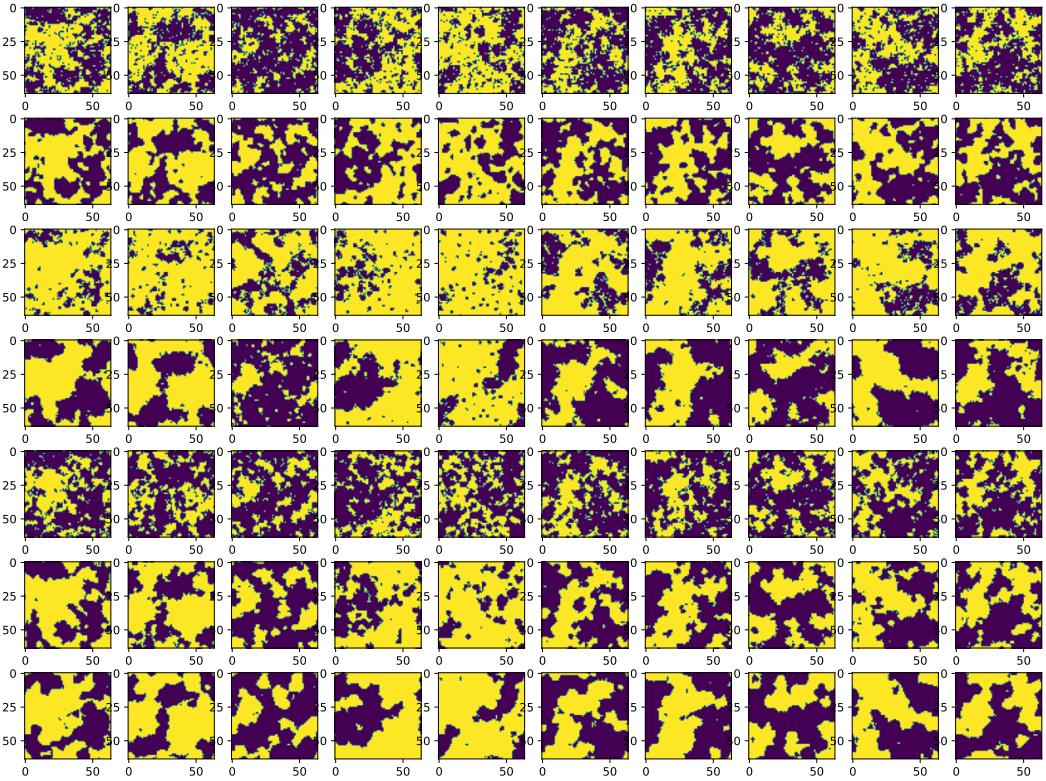


Figure A.37: Reconstructed model for $\beta = .415$.

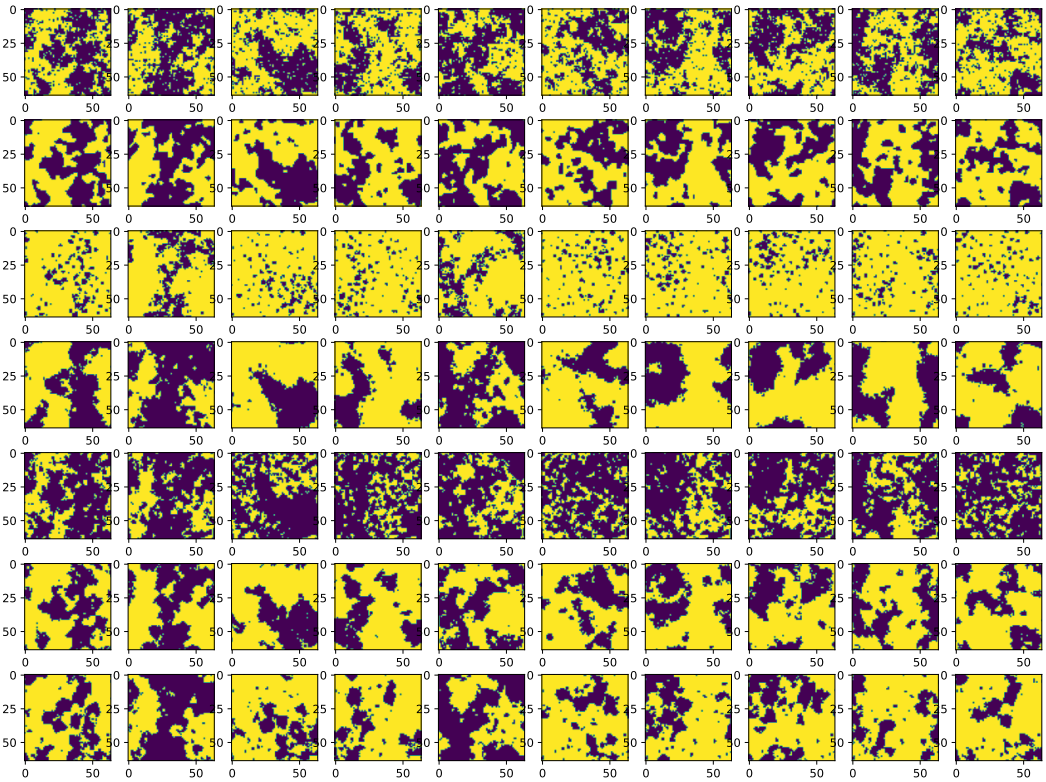


Figure A.38: Reconstructed model for $\beta = .42$.

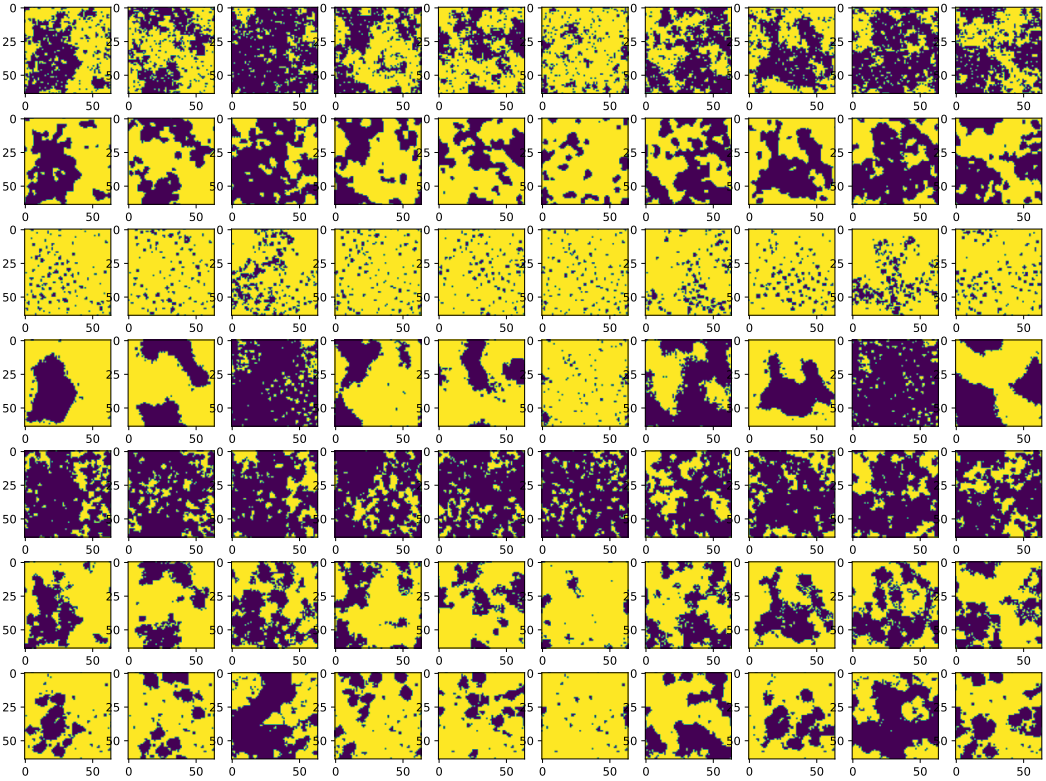


Figure A.39: Reconstructed model for $\beta = .425$.

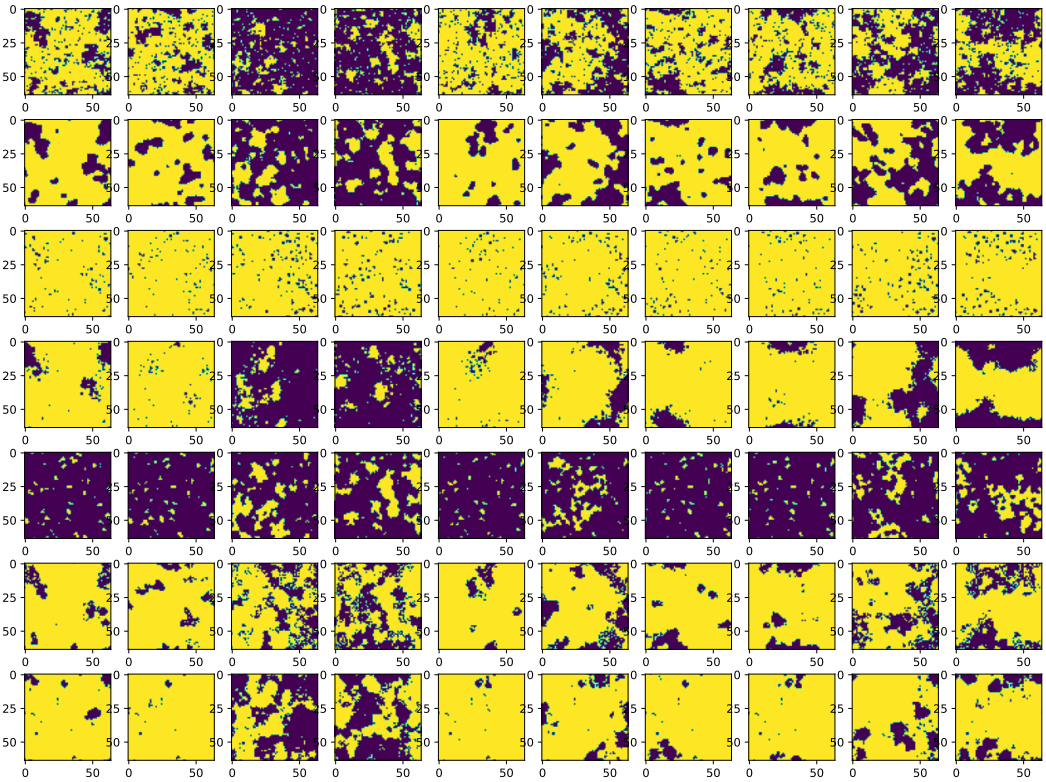


Figure A.40: Reconstructed model for $\beta = .43$.

Appendix B

RENORMALIZATION TECHNIQUES AND DEEP LEARNING PLOTS

B.1 Generative Model Plots

This section contains plots showing the generative models as discussed in Section 5.1. The first set of plots contains plots of the models themselves, with the first row containing the original 8 by 8 lattice models and the second row containing the 64 by 64 generated model. The second set of plots consists of fits to the correlation function for each model, in order to derive the correlation lengths.

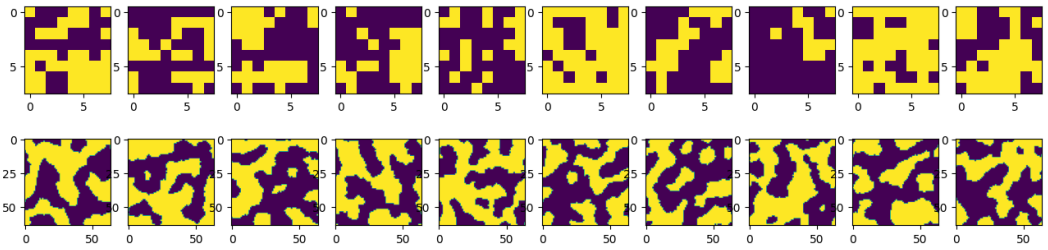


Figure B.1: Generative model for $\beta = .395$.

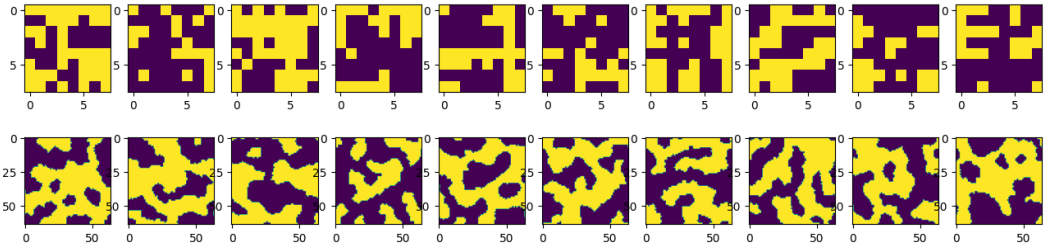


Figure B.2: Generative model for $\beta = .4$.

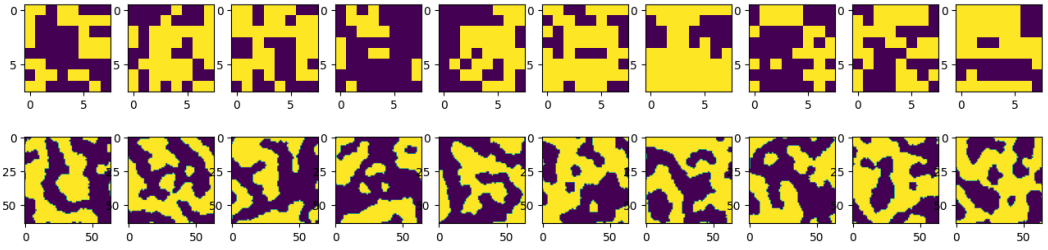


Figure B.3: Generative model for $\beta = .405$.

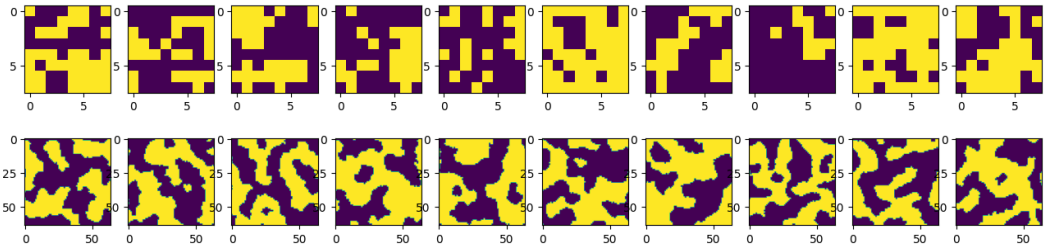


Figure B.4: Generative model for $\beta = .41$.

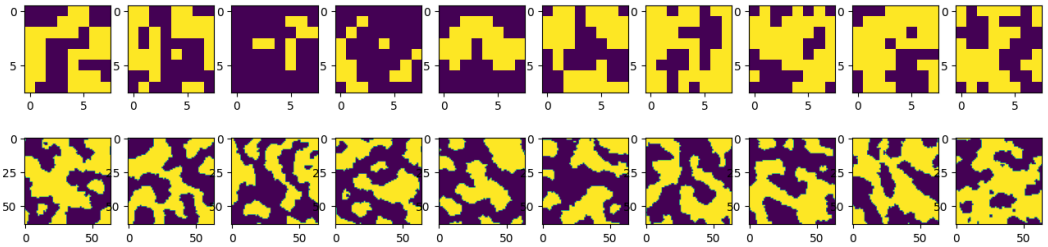


Figure B.5: Generative model for $\beta = .415$.

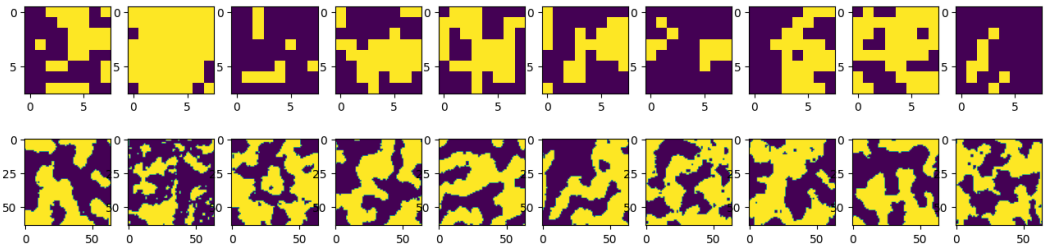


Figure B.6: Generative model for $\beta = .42$.

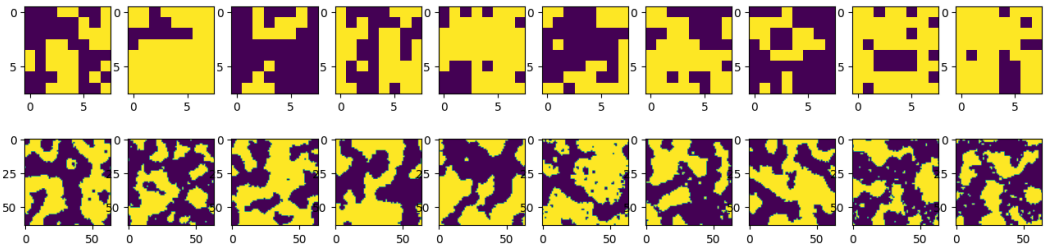


Figure B.7: Generative model for $\beta = .425$.

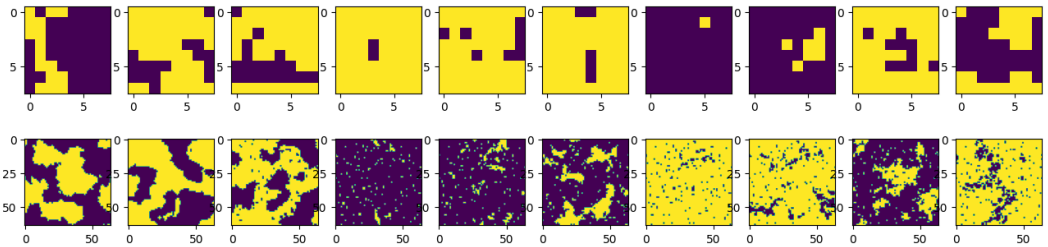
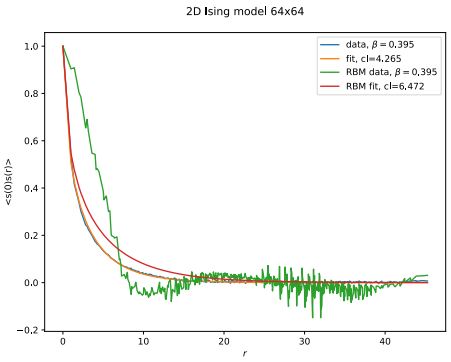
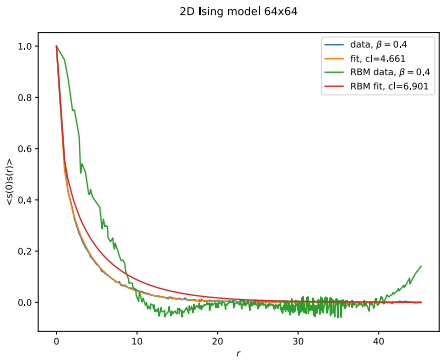


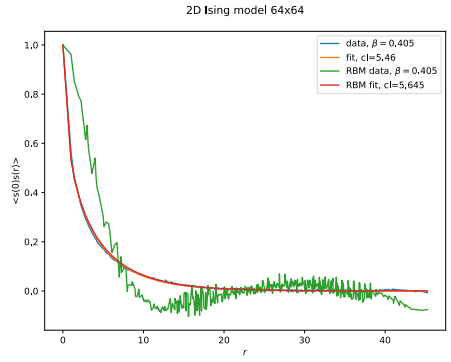
Figure B.8: Generative model for $\beta = .43$.



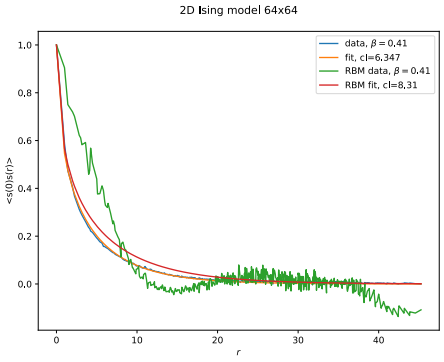
(a) $\beta = .395$.



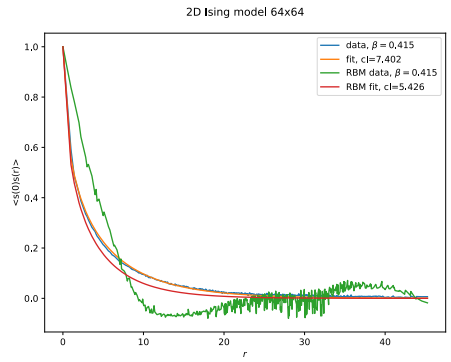
(b) $\beta = .4$.



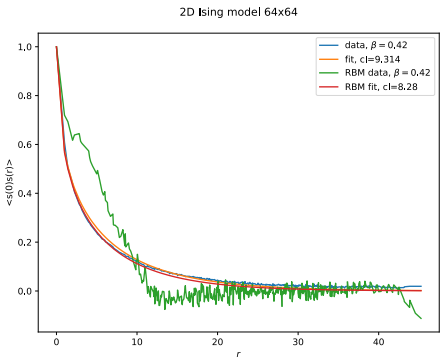
(c) $\beta = .405$.



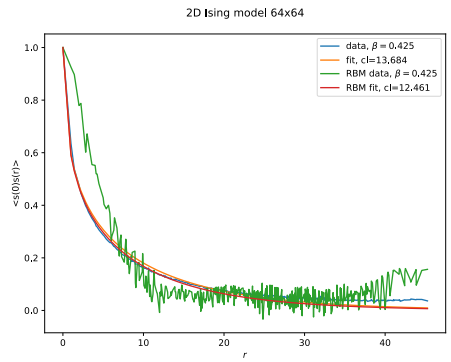
(d) $\beta = .41$.



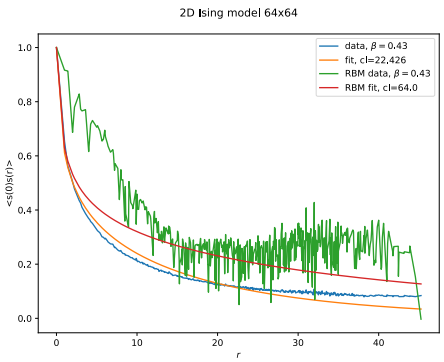
(e) $\beta = .415$.



(f) $\beta = .42$.



(g) $\beta = .425$.



(h) $\beta = .43$.

Figure B.9: Correlation length comparisons for generative model.

B.2 Ising Spin Plots

This section contains plots showing the RBM spin models as discussed in Section 5.2. The first set of plots contains the models themselves, with the first row containing the original Wolff generated model, and the next three rows consisting of the RBM coarse-grained modes. The second set of plots consists of fits to the correlation function for each model, in order to derive the correlation lengths.

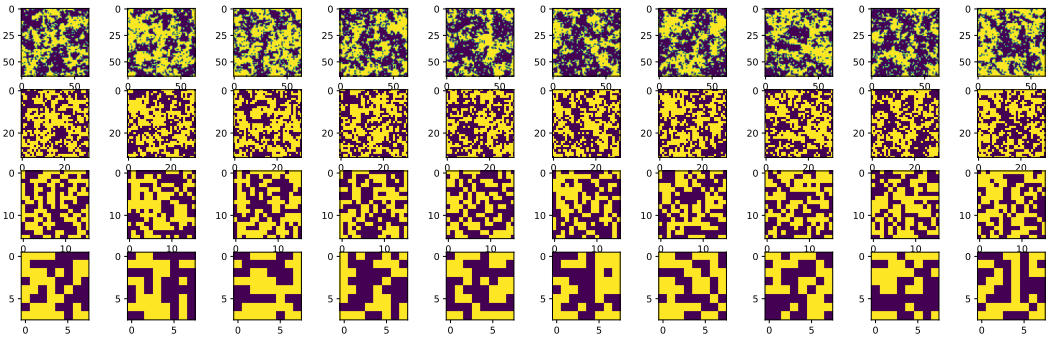


Figure B.10: RBM coarse graining for $\beta = .395$.

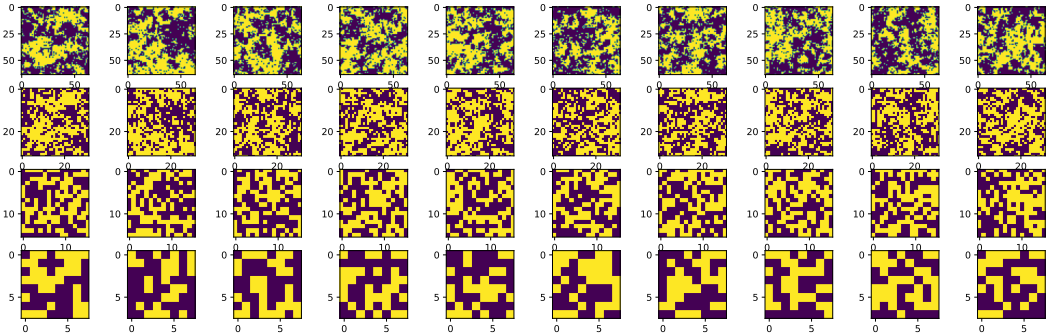


Figure B.11: RBM coarse graining for $\beta = .4$.

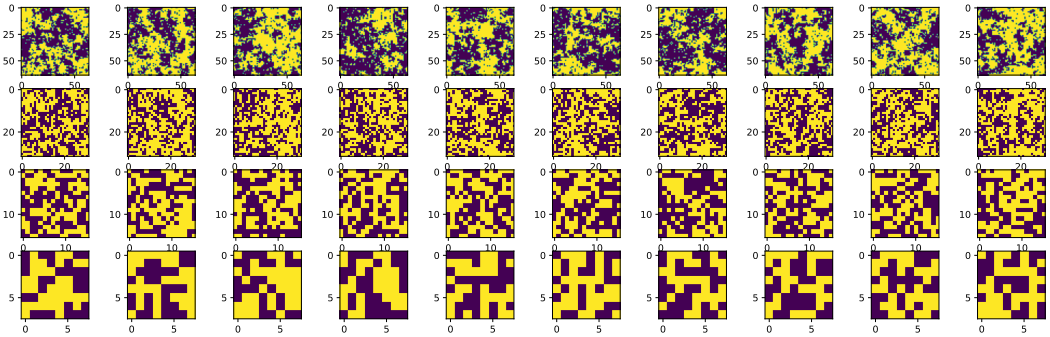


Figure B.12: RBM coarse graining for $\beta = .405$.

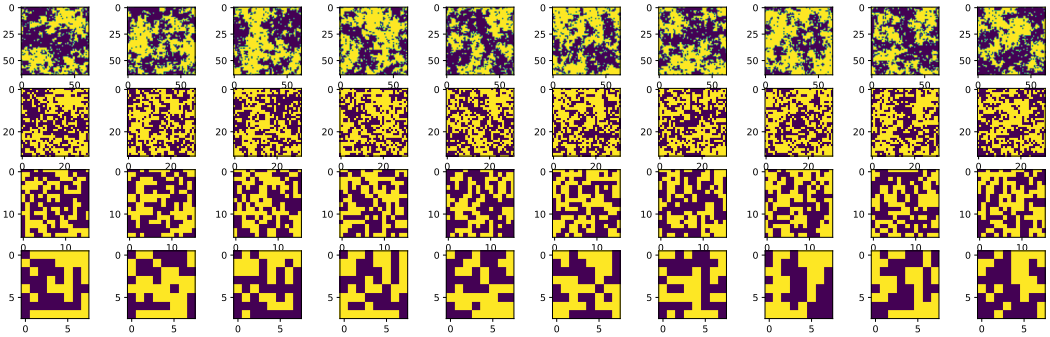


Figure B.13: RBM coarse graining for $\beta = .41$.

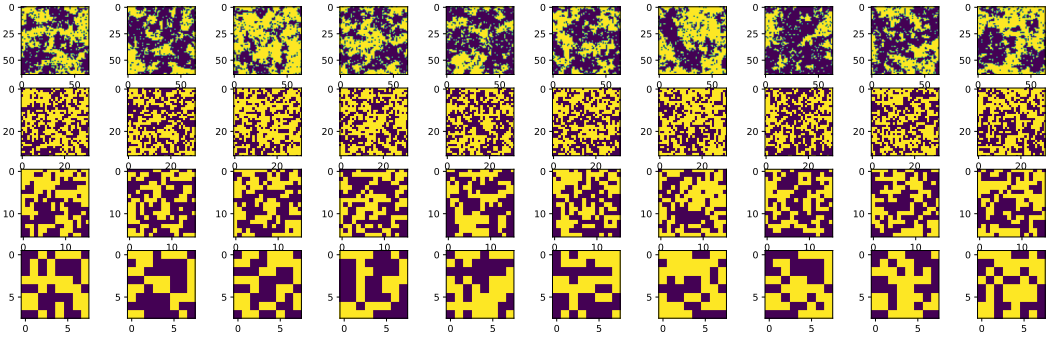


Figure B.14: RBM coarse graining for $\beta = .415$.

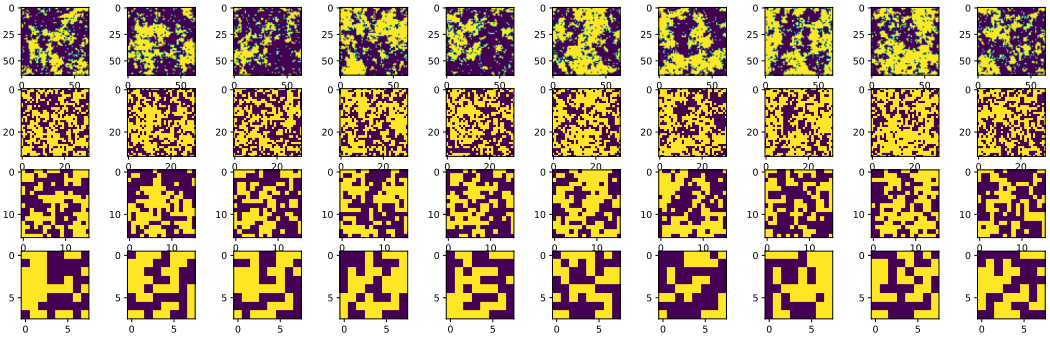


Figure B.15: RBM coarse graining for $\beta = .42$.

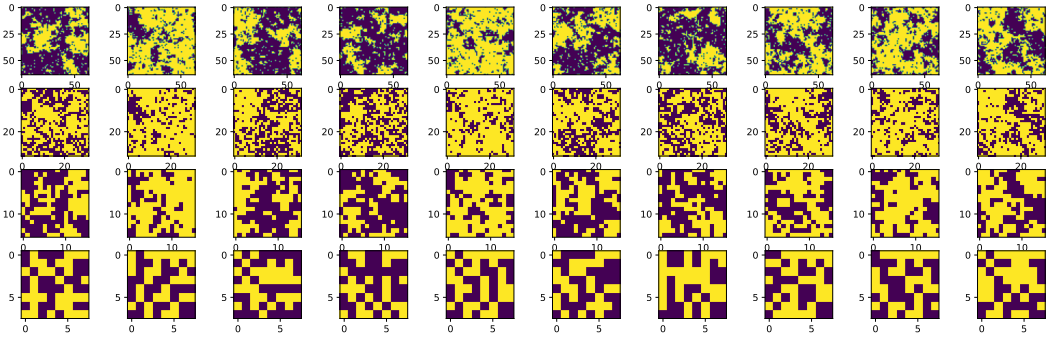


Figure B.16: RBM coarse graining for $\beta = .425$.

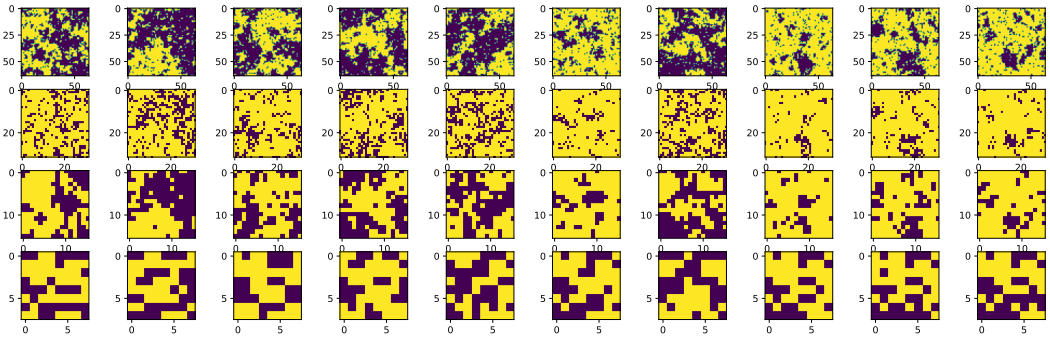
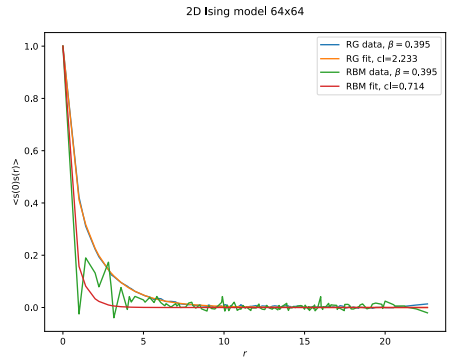
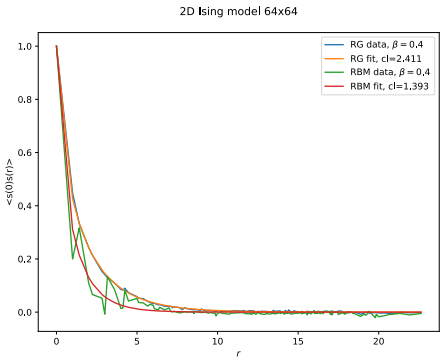


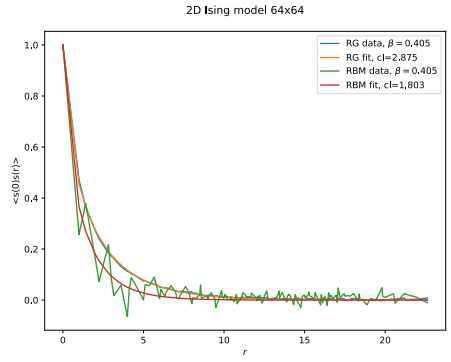
Figure B.17: RBM coarse graining for $\beta = .43$.



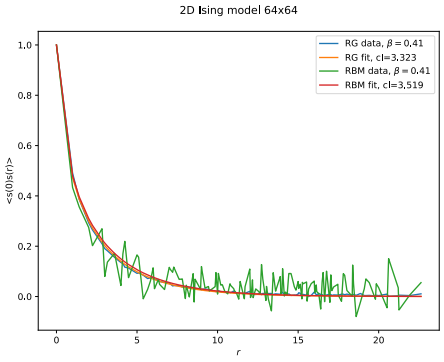
(a) $\beta = .395$.



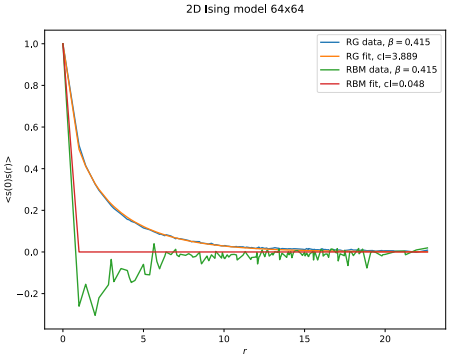
(b) $\beta = .4$.



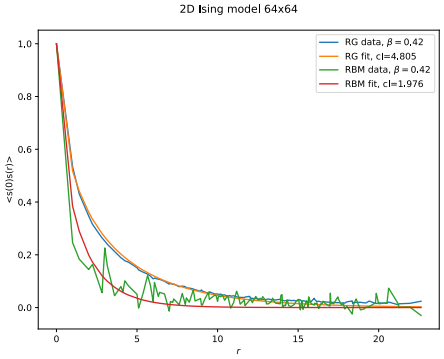
(c) $\beta = .405$.



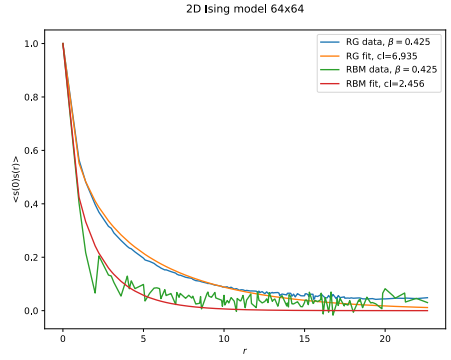
(d) $\beta = .41$.



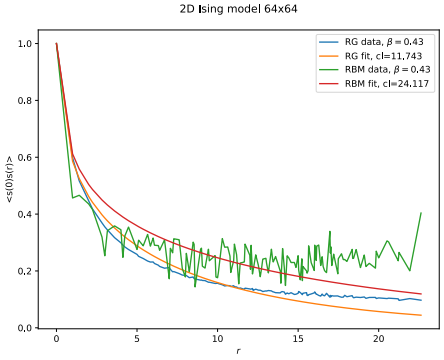
(e) $\beta = .415$.



(f) $\beta = .42$.

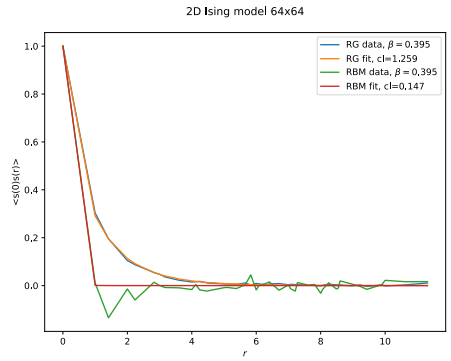


(g) $\beta = .425$.

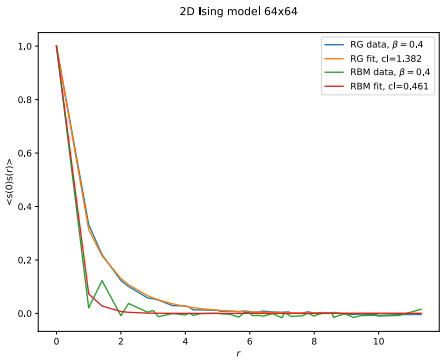


(h) $\beta = .43$.

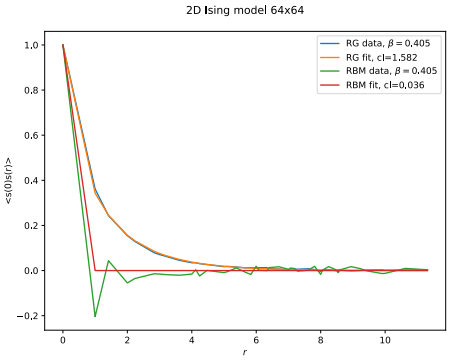
Figure B.18: Correlation length comparisons for Layer 1.



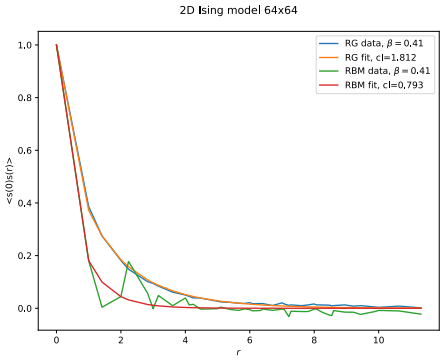
(a) $\beta = .395$.



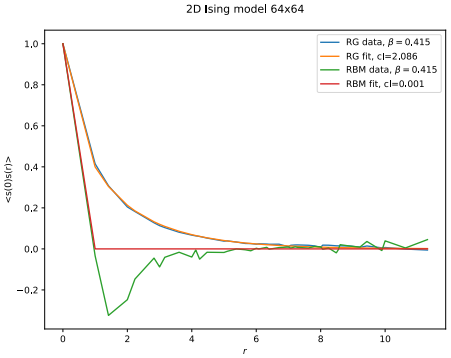
(b) $\beta = .4$.



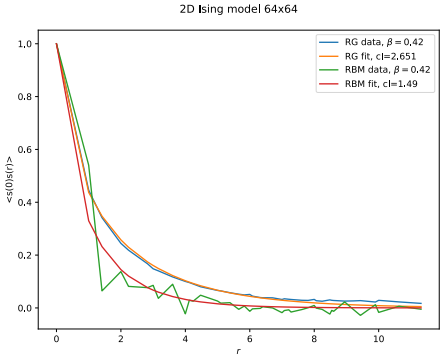
(c) $\beta = .405$.



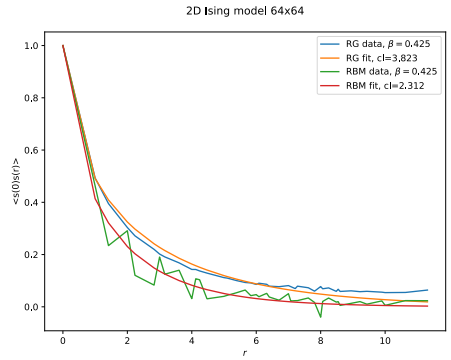
(d) $\beta = .41$.



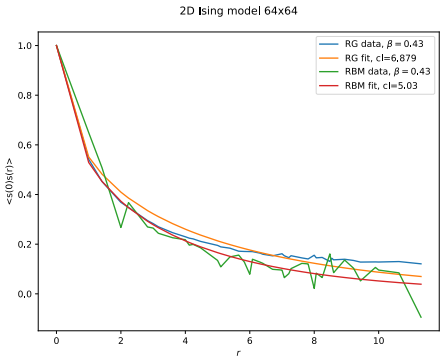
(e) $\beta = .415$.



(f) $\beta = .42$.

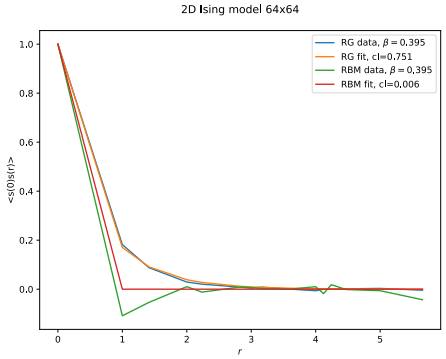


(g) $\beta = .425$.

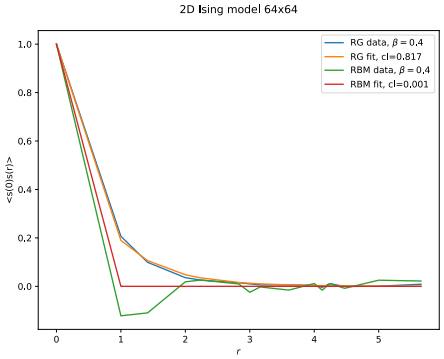


(h) $\beta = .43$.

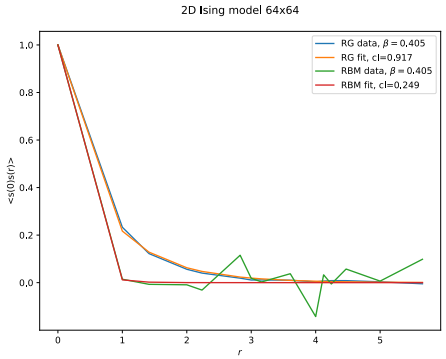
Figure B.19: Correlation length comparisons for Layer 2.



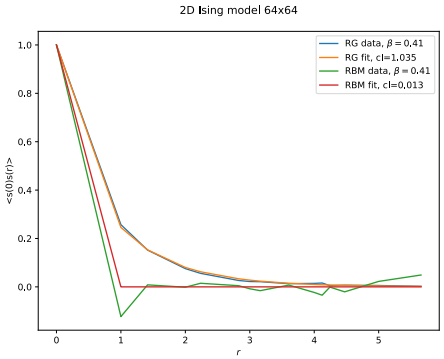
(a) $\beta = .395$.



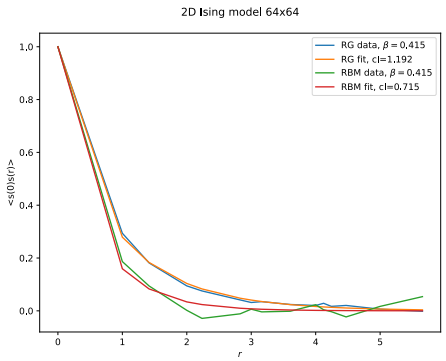
(b) $\beta = .4$.



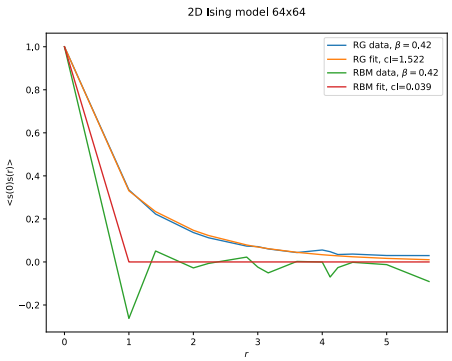
(c) $\beta = .405$.



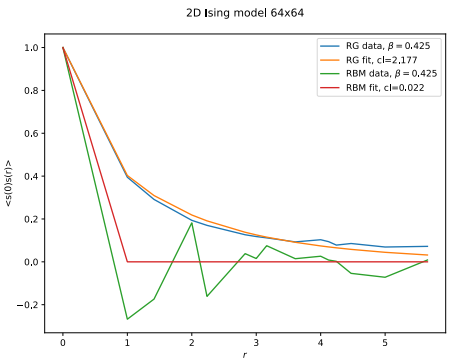
(d) $\beta = .41$.



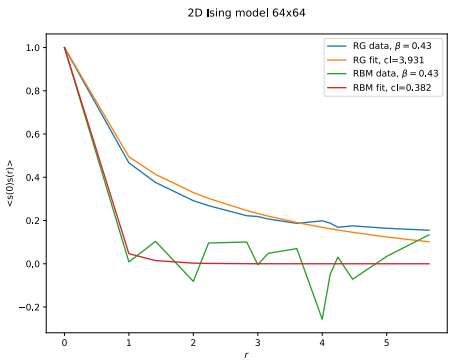
(e) $\beta = .415$.



(f) $\beta = .42$.



(g) $\beta = .425$.



(h) $\beta = .43$.

Figure B.20: Correlation length comparisons for Layer 3.