A coarse graining perspective of deep learning

Ellen de Mello Koch

Supervisor: Prof. Ling Cheng

Department of Electrical and Information Engineering
University of the Witwatersrand

This thesis is submitted for the degree of
Doctor of Philosophy

April 2021

To my loving mother, who has always believed in me and given me the world.
To my incredible husband and best friend who supports me in all endeavors.
To my older brothers for their love, guidance, support and friendship.
To Robert, Lily, Anita and Roger for showing me the true value of family.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Ellen de Mello Koch
April 2021

Acknowledgements

I would like to express my gratitude to my supervisor, Professor Ling Cheng, who has provided support and guidance which extended well beyond the work presented here.

I wish to acknowledge the support and guidance provided by my brother, Robert de Mello Koch. His willingness to give his time so generously and share his ideas has been very much appreciated.

Abstract

Deep learning has shown remarkable success at a number of tasks in various applications. This success is well appreciated in practice but theoretical explanations for how deep learning works are in their infancy. Previous work suggests that a link may exist between deep learning and the renormalization group (RG). RG is a powerful tool with a strong theoretical foundation. It allows one to move from an enormous set of microscopic parameters which describe a system to a succinct macroscopic description. RG achieves this by repeatedly performing a local coarse graining. Deep learning can also be viewed as performing a form of averaging or coarse graining. Deep networks reduce a large number of inputs to a smaller number of outputs which summarize the input training data. This suggests that a link between RG and unsupervised deep learning may exist. Such a link would give a possible starting point from which to develop a theoretical framework for deep learning by exploiting the rich theory of RG. This thesis is focussed on establishing this link in a quantitative way.

The studies presented here are carried out on models of a magnet, namely the Ising model and long range spin chain, as well as the MNIST (Modified National Institute of Standards and Technology - a handwritten digit dataset), TensorFlow Flowers and SWIMSEG (Singapore Whole sky IMaging SEGmentation - a sky and cloud image dataset). We extend current work by probing the restricted Boltzmann machine (RBM) flow fixed point using two-point spin correlators. In the case of the Ising model, the RBM does not correctly reproduce properties related to longer ranged scaling dimensions. For the long range spin chain, the RBM does not correctly reproduce even the shortest distance scaling dimensions. This is an interesting result as this suggests RBMs are better at learning local features as opposed to long ranged interactions.

We introduce a method for comparing the coarse graining of RBMs and RG using correlation functions between inputs and outputs of both transformations. By studying spatial correlators of the input output correlators we learn about the local patterns each output node encodes. Locality is a key characteristic of RG which motivates this method. Another key characteristic of the RG flow is the flow of temperature. We compare the flow of temperature in stacked RBM networks to several steps of RG. Here we see RG-like behaviour when

studying the Ising model but when there are longer ranged interactions as in the case of the long range spin chain this no longer holds.

The final contribution relates to the generalization and noise rejection properties of RBMs and autoencoders. One of the main results we present shows that unsupervised deep learning is performing a coarse graining of the momentum space RG near the free field fixed point. By performing an eigen decomposition of the trained weight matrix we show that for large singular values the hidden singular vector is obtained by discarding the high Fourier modes of the visible singular vector. These large eigenvalues and associated visible and hidden singular vectors define the relevant modes of the training data. We also find strong agreement between the subspaces defined by the training data covariance matrix and the trained weights. Using an initial condition related to the training data covariance matrix we find training times improve by a factor between 4 and 100. This is shown not only on models of magnets but also using the MNIST, TensorFlow Flowers and SWIMSEG datasets.

Deep learning when viewed as curve fitting results in the generalization puzzle. There are millions of parameters which must be fitted using only tens or hundreds of thousands of training samples. The fact that deep networks do generalize goes against logic as we require more samples than parameters to fit the training data. The result we present provides a different framework in which to view unsupervised deep learning and shows that only a handful of parameters are needed to capture the relevant degrees of freedom. We show that deep networks are able to learn the underlying structure present in the training data and many of the parameters are irrelevant and can be set to 0. The reduction is so large that the number of training samples is greater than the number of parameters which need tuning. This result is a promising step forward for understanding unsupervised deep learning. In addition we can now phrase new questions in terms of a coarse graining framework. Future work includes extending these results to supervised networks.

Table of contents

List of figures

List of tables

List of Symbols

Mathematical Symbols

Δ	scaling dimension
$\langle X \rangle$	expectation value of X
σ	lattice spin
$b^{(h)}$	hidden bias
$b^{(v)}$	visible bias
h_a	hidden node
J	coupling constant
k	Boltzmann constant
m	mass
$p(\mathbf{v}, \mathbf{h})$	probability of hidden and visible vector
T_c	critical temperature
v_i	visible node
W	weight matrix
π	$\simeq 3.14\ldots$

Acronyms / Abbreviations

CFT	Conformal field theory
FFT	Fast Fourier transform

IR Infrared

MC Monte Carlo

MCMC Markov Chain Monte Carlo

MNIST Modified National Institute of Standards and Technology

RBM Restricted Boltzmann machine

RG Renormalization group

RGM Renormalization group machine

SWIMSEG Singapore Whole sky IMaging SEGmentation

UV Ultraviolet

Chapter 1

Introduction

In recent years there has been a renewed interest in machine learning. This is in large due to an increase in compute power, a reduction in storage costs and an increase in the quantity and availability of data. Specifically deep learning has shown remarkable success by outperforming humans at a wide range of tasks such as Go [1], the Atari games [2] and image classification and recognition tasks [3]. The increase in compute power has made it feasible to train deeper networks. These deep architectures have an enormous number of parameters which require tuning during training. In the case of top performing image classification networks the number of parameters is in the order of millions with AlexNet [4] having approximately 61,000,000 parameters and GoogleNet approximately 7,000,000 parameters [5]. Despite the ability to train these complex networks and their incredible performance, a theory explaining deep learning is lacking.

No formal scientific definition exists at present for what learning is. We know that humans are good at learning. People can accurately identify an object in nature as a tree. A tree is made up of many features such as leaf size, leaf colour, bark texture and height. Trees may have different branch configurations and go through seasonal changes. Even with all of this variation, people can agree upon what objects are trees, bushes or grass. There are certain aspects of individual trees which will be common amongst all trees. These features do not change a lot when we look at different trees. An example of such a feature may perhaps be the fact that all trees have bark. Features which may show large variations between different types of trees could be height, leaves and bark texture. Even a particular tree may look different depending on the time of year that it is observed. These features vary between different types of trees, however, are not important when labelling an object as a tree. We sift through all these features to arrive at an object label, discarding some smaller unimportant details. This idea of sorting through features and finding which features are important to aggregate, which can be described as a type of coarse graining, is one possible description of

how learning works. Coarse-graining means to go from a very detailed, fine level description to a less detailed coarse description.

Many people have discussed the idea that deep learning can be viewed as performing a type of coarse-graining [6–8]. Although these papers are focussed on specific applications, this theme of data being coarse-grained is evident. A reasonable hypothesis is that learning is coarse-graining.

In physics there exists a powerful and well understood coarse-graining tool, the renormalization group (RG). RG is used to move from a microscopic to a macroscopic description of a system by repeatedly applying steps of coarse graining. This is achieved by removing high frequency modes which represent the irrelevant noise in the data, and only keeping the low frequency modes which represent the signal of interest. Imagine a swimming pool filled with an enormous number of water molecules. Each molecule has its own position, mass and momentum and so there exist a vast number of microscopic parameters which characterize the system. Often we are interested in the average properties of the water molecules as a whole. These properties may include the pressure or volume of the body of water. RG gives us a systematic way to perform the coarse graining in order to obtain a macroscopic description given the microscopic description.

Some studies have suggested that a link between RG and deep learning may exist [9–11]. If this link does exist we could rely on the rich theoretical framework provided by RG to understand deep learning. This thesis is aimed at examining this link.

Below the background is sketched after which the problem statement and research objectives are outlined. The chapter ends with a brief summary of the contributions made by this work. Each objective and contribution includes a reference to where this is included in the thesis.

1.1 Background

At present, explanations for how and why deep learning works are in their infancy. In [9] a suggestion is made that deep learning and RG are equivalent. The authors provide an identity that they claim proves an equivalence between the two coarse graining methods. However a crucial equation upon which their argument rests does not provide a unique characterization for RG and the RBM [12] (this point is discussed in Chapter 2).

The link is pursued further in [10, 11] where authors study a model of a magnet, the Ising model. The two dimensional Ising model consists of a two dimensional grid of lattice sites where each site contains a spin taking values of ± 1. This model is well studied and understood in physics and is therefore a useful setting in which to probe the link between RBMs and

RG. In addition, the Ising model only allows couplings between nearest neighboring spins. This enforces locality in the data which is similar to the locality found in images. In images nearby pixels are likely to have a similar colour and pixels far away from each other are less likely to be similar. It is well known that deep networks perform well on image data and the Ising model provides an analogous setting.

To generate data for the Ising model, Monte Carlo simulations are used. The authors in [10] claim that the RBM learns the critical temperature of the Ising model. This is demonstrated by generating RBM flows of a single network and measuring their temperatures using a supervised neural network. The temperature measurements show that the RBM flows converge to a temperature very close to the critical temperature. In addition to this they provide analysis of the off diagonal components of the weight matrices showing that as the distance from the diagonal increases, so the values decrease. They show that for high temperatures, the fall off is faster than at low temperatures which is consistent with what we expect from the spin model. At high temperatures interactions are more local than at low temperatures.

In [10, 11] only single-layered RBM networks are studied. Using the trained network parameters a flow of inputs and outputs are generated by the network. In RG we perform a number of steps to eventually arrive at a coarse grained description. If RBMs do perform an RG-like coarse graining, a network with multiple layers must be investigated as this is one way we may be able to find similarities between the two techniques.

In studies aimed at understanding noise rejection properties of networks [13, 14], authors have studied the eigen decomposition of trained fully connected weight matrices. Results show that many of the singular values are so small they can be set to 0. This results in a large reduction in the number of parameters. The authors show that this handful of large singular values is linked to noise rejection properties of the network. This is not surprising as only the few singular vectors associated to large singular values will carry the noise to the output. This observation however needs to be explored further to gain understanding.

1.2 Problem statement

A theory for deep learning is yet to be developed. Understanding how and why deep neural networks perform so well would give a concrete framework in which to ask important questions about this tool. Such questions would relate to network architecture, training speeds, or understanding in which settings these methods fail. Being able to answer questions such as these in a precise manner could aid in further development and refinement of deep learning.

1.3 Research objectives

The research objectives include:

- To develop a quantitative measure which probes similarities/differences in both coarse graining settings. We achieve this by studying statistical properties of the transformations to determine whether the two behave similarly. This is explored in Chapter 2.
- To probe the link between a single RBM's flow and the RG flow. We compare statistical properties of spin configurations generated by a trained RBM using spatial correlators to probe whether the RBM flow converges to the critical point of these spin systems. This analysis is performed on the two dimensional Ising model (in Chapter 2) as well as the long range spin chain (in Chapter 3).
- To study the link between stacked (more than one layer) RBM flow and the RG flow. This analysis is again performed on the two dimensional Ising model (in Chapter 2) as well as the long range spin chain (in Chapter 3). Further, by performing the singular value decomposition of a block spin coarse graining rule with overlapping blocks and the trained weights of the RBM, we cement the connection (in Chapter 4).
- To study properties of unsupervised networks related to noise rejection and generalization. This analysis is concerned with RBMs and autoencoders trained on Ising, MNIST, TensorFlow Flowers and SWIMSEG datasets (in Chapter 4).

1.4 Contributions and layout of this thesis

This thesis is organized in order of the publications produced as follows

1. Chapter 2 - Is unsupervised deep learning a renormalization group flow? [15]
2. Chapter 3 - Short sighted deep learning [16]
3. Chapter 4 - Why Unsupervised Deep Networks Generalize [17]

The main contributions of this thesis are summarized as follows with reference to where they appear.

1.4.1 Quantitative statistical measure to compare RG and RBMs

In Chapter 2 we develop an input output correlation function which can be used to study patterns between input and output nodes. We find that the RBM learns similar local patterns to the RG where a cluster of inputs are encoded in a specific output node.

We further quantify these input output correlators by calculating spatial correlators. The spatial correlators give us important information about the locality encoded in the input output correlators. Locality is a signature characteristic of RG so is useful to study.

1.4.2 The RBM flow versus the RG flow

We explore the RBM flow in settings of the Ising model and long range spin chain. For the Ising model the RBM flows correctly reproduce the spin scaling dimension but not the energy scaling dimension. This is discussed in Chapter 2. For the long range spin chain the RBM fails to correctly reproduce all spin correlator scaling dimensions. This is discussed in Chapter 3.

This suggests that RBMs represent the input data more faithfully when interactions are local. It also confirms that the RBM does not always learn critical points of spin models.

1.4.3 Stacked RBM flows vs RG flows

When we study the stacked RBM flow in relation to the RG flow we find that in the Ising model setting, both coarse graining schemes show an increase in temperature as we progress through the flow. This is discussed in Chapter 2.

However in the case of the long range spin chain we do not see the expected RG behavior in the data generated by the RBM. Here the RBM shows a decrease in temperature as we move through the layers of the network. This is discussed in Chapter 3.

These results again confirm that the RBM is not good at learning when the data has long ranged interactions and is more suited to learning localized patterns.

1.4.4 Explaining why deep networks generalize

We demonstrate through numerical experiments that RBMs and autoencoders are performing a coarse graining of the momentum space RG near the free field fixed point. From this result we can see that the correct way to understand unsupervised deep learning is from a coarse graining perspective as opposed to curve fitting.

We achieve this result by performing an eigen decomposition of trained weight matrices and studying the visible and hidden singular vectors. The visible and hidden singular vectors

which correspond to large singular values, only have support at low frequency Fourier modes. These low frequency modes relate to coarse, relevant features of the data. We show that by truncating the high Fourier modes of the visible singular vectors, we obtain the hidden singular vectors. This result shows a dramatic reduction in the number of parameters which are relevant to learning the input data. In addition to this we demonstrate that by initializing the weights to the subspace of the data covariance matrix we see training speeds improve by between 4 and 100 times. This is a key result that opens up a vast range of new questions within the coarse graining setting. This is discussed in Chapter 4.

Chapter 2

Is deep learning an RG flow?

2.1 Introduction

The power of machine learning and artificial intelligence is established: these are powerful methods that already outperform humans in specific tasks [18–21]. Much of the research carried out in machine learning is of an applied nature. It establishes the practical utility of the method but does not construct an understanding of how deep learning works or even if such an understanding is possible [22–27]. Consequently, deep learning remains an impressive but mysterious black box. A possible starting point for a theoretical treatment suggests that deep learning is a form of coarse graining [20, 28, 29]. Since there are more neurons than output neurons this is almost certainly true. The real question is then if this is a useful observation, one that might shed light on how deep learning works. This is the question we take up in this thesis.

We argue that understanding unsupervised deep learning as a form of coarse graining is a useful observation, and make the case by adopting and adapting several ideas from theoretical physics. Specifically, in theoretical physics there is a sound framework to carry out coarse graining, known as the renormalization group (RG) [30]. RG provides a systematic way to construct a theory describing large scale features from an underlying microscopic description, which can be understood as recognizing sophisticated emergent patterns, a routine achievement of deep learning. Further, RG is applicable to field theories, that is, to systems with a large number of degrees of freedom so that it seems that RG is well positioned to deal with massive data sets. Finally, the way in which RG works is, in contrast to deep learning, well understood and can be described in precise mathematical language. These features suggest that RG may provide a useful framework in which to describe deep learning and attempts to argue that this is the case have been made in [9–11, 31]. We focus on unsupervised learning by a restricted Boltzmann machine (RBM).

Two distinct possible connections to RG have been attempted, both relevant to our study. The first [9] is an attempt to link unsupervised deep learning to the RG flow. The RG flow is a smooth process during which degrees of freedom are continuously averaged out, so that we flow from the initial microscopic description to the final macroscopic description. In deep learning one stacks layers of networks to obtain a deep network. The proposed connection of [9] suggests that each layer in the stack performs a small step along the RG flow[1]. We contribute to this discussion by developing quantitative tools with which this proposal can be explored with precision. The basic objects that appear in our analysis are correlation functions between the visible and the hidden neurons. This allows us to decode the mechanics of the RBM's pattern generation and to compare it to what the RG is doing. Although there are important differences, our results indicate remarkable similarities between how the RBM and RG achieve their results. The second approach [10, 11] builds an RBM flow using the weight matrix of the neural network after training is complete. The results of [10, 11] suggest that the RBM flow is closely related to the RG flow. We carry out a critical examination of this conclusion. The central tools we employ are correlation functions defined using the patterns generated by the RBM. We give a detailed and precise argument showing that the largest scale features of RG and RBM patterns are in complete agreement. The correlation functions involved are non-trivial probes of the statistics of the generated pattern[2] so the conclusion we reach is compelling. We also find that if one probes smaller scale features there are important differences between the two patterns. We will comment further on the interpretation of these results in the conclusions.

At this point, a comment is in order. The word "deep" in "deep learning" indicates that many layers are stacked to produce the network. Each layer in our network is an unsupervised RBM. The word "flow" in "RG flow" indicates that many small steps of coarse graining are carried out. Each step performs a local averaging and one basic step is being repeated. In this study we are developing methods that allow a comparison between one step of the renormalization group flow to one layer of the deep network. Thus, although we spend much of our effort on comparing a single RBM layer to a single step in the RG flow, we are interested in understanding the learning achieved through the composition of many layers as an RG flow. It is in this sense that although we study a single layer we nevertheless claim we are exploring the problem of deep learning.

The setting for our study is the two dimensional Ising model [33]. This is a simple model of a magnet, built for many individual "spins" each of which should be thought of as a microscopic bar magnet. Each spin can be aligned "up" or "down". Spins align at low

[1] For a critical discussion of this proposal, see [29, 32]

[2] The studies carried out in [10, 11] used averages of the RBM pattern. Our correlation functions provide more sensitive probes into the structure of the pattern.

temperatures producing a magnet. At high temperatures, spins are aligned randomly and there is no net magnetic field. The spins themselves define a binary pattern (the two states are up or down) and it is these patterns that the RBM learns. An important motivation for this choice of model is that it is well understood. The theory exhibits a first order phase transition terminating at a critical point. The theory at the critical point enjoys a conformal symmetry so that it can be solved exactly. It exhibits many interesting observables which we use to explore how deep learning is working [34, 35]. For example, if a neural network generates a microstate of the model, we can ask what the corresponding temperature of the microstate is. At the critical point special observables known as primary operators, can be defined. Their correlation functions are power laws with powers that are known. These are the natural variables which encode, completely, the long scale features of the patterns. In this way, the Ising model gives a framework to explore unsupervised deep learning both through the results of numerical experiments and using the complete understanding of the large scale features of the coarse grained system. To probe whether unsupervised deep learning is a type of coarse graining, we will see that this knowledge of correlations on large length scales is a valuable tool.

Our study of an Ising magnet may seem rather far removed from more usual (and practical) applications, including for example image recognition and manipulation. However, one might be optimistic that lessons learned from the Ising model are applicable to these more familiar examples. Indeed, the energy function of the Ising model tries to align nearby spins with the result that nearby spins are correlated. This is not at all unlike an image for which the color of nearby pixels is likely to be correlated [36].

A description of deep learning in the RG framework would have important implications. RG explains how macroscopic physics emerges from microscopic physics. This understanding leads to an organization of the microscopic physics into features that are relevant or irrelevant, so that in the end the emergent patterns depend only on a small number of relevant parameters. Carried over to the deep learning context, a similar understanding will strive to explain what features of the data and which weights in the network are important for deep learning. Such an understanding would have implications for what architectures are optimal and how the learning process can be improved and made more efficient.

We now sketch the content of the chapter and outline how it is organized. In Section 4.1 we give a quick review of RBMs, RG and the Ising model, providing the background needed to follow subsequent arguments. In Section 2.3 we consider the RBM flows defined using the matrix of weights learned by the network [10, 11]. By studying correlation functions of primary operators of the Ising conformal field theory, we argue that although the RBM and RG patterns agree remarkably well on the largest scales of the pattern they differ on

the short scale structure. In Section 2.4 we examine the possibility that unsupervised deep learning reconstructs an RG flow, with each layer of the deep network performing one step of the flow. Our discussion begins with a critical look at the argument given in [9] that claims that unsupervised deep learning is mapped onto the RG flow. The argument shows a system of equations that is obeyed by both the RBM and a variational realization of the RG flow. Our basic conclusion is that the argument of [9] only shows that aspects of the RBM learning are consistent with the structure of the RG transformation. Indeed, we explicitly construct examples that satisfy the equations derived by [9] that certainly do not perform an RG flow or arise from an RBM. Nevertheless, the arguments of [9] are compelling and we find the possible connection between unsupervised deep learning and RG fascinating and deserving of further study. Towards this end we couch some of the qualitative observations of [9] as statements about the behavior of well chosen correlation functions. The form of these correlators, puts certain qualitative observations of Section IV.B. of [9] onto a firm quantitative footing. Finally we study the RG flow of the temperature. This turns out to be interesting as it reveals a further difference between the RBM patterns and RG. In the final Section of this chapter, we discuss our results and suggest open directions that can be pursued.

2.2 RBM, RG and Ising

In this section we introduce the background material used in our study. The first subsection reviews RBMs emphasizing both the structure of the network and its implementation. Following this, the RG is reviewed, with an emphasis on aspects relevant to unsupervised deep learning. This section concludes with a review of the Ising model, motivating why the model is considered.

2.2.1 Restricted Boltzmann Machines

RBMs perform unsupervised learning to extract features from a given data set [37–39]. They have a visible (input) and a hidden (output) layer. The visible layer is made up of visible nodes, v_i with $i = 1, 2 \cdots, N_v$ and the hidden layer is made up of hidden nodes, h_a with $a = 1, 2, \cdots, N_h$ as illustrated in Figure 2.1.

The visible nodes are set with values of ± 1 and the trained network generates a corresponding pattern by setting the output nodes to ± 1. The values of the hidden neurons are obtained by evaluating a non-linear function on a linear combination of the visible neurons, perhaps offset by a constant bias. The nonlinear function we use here is the hyperbolic

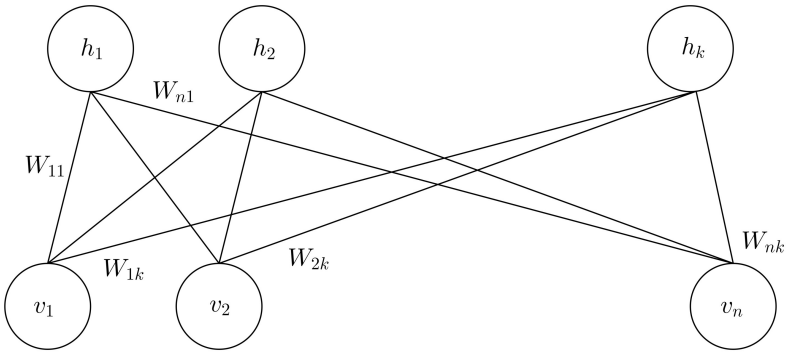


Fig. 2.1 An RBM network with visible nodes v_i and hidden nodes h_a where $N_v = n$ and $N_h = k$. Connections between visible and hidden nodes are each associated with a weight W_{ia}.

tangent which can be seen in equation (2.12). The specific linear combination of neurons is represented by connections between nodes, with a weight for each connection. For the RBM there are connections between every visible node and every hidden node, while nodes belonging to the same layer are not connected. The "unrestricted" Boltzmann machines allow connections between any two nodes in the network [40], but this generality comes at a cost: training algorithms are much less efficient [37–39]. The connection between visible node v_i and hidden node h_a is assigned a weight, W_{ia}, and visible (hidden) nodes are assigned a bias $b_i^{(v)}$ ($b_a^{(h)}$). Using these ingredients we define a Hamiltonian for the RBM

$$E = -\sum_a b_a^{(h)} h_a - \sum_{i,a} v_i W_{ia} h_a - \sum_i b_i^{(v)} v_i, \tag{2.1}$$

where $h_a, v_i \in \{-1, 1\}$. The RBM defines the probability distribution for obtaining configurations $\mathbf{v}$ and $\mathbf{h}$ of visible and hidden vectors by [41]

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathscr{Z}} e^{-E}, \tag{2.2}$$

where $\mathscr{Z}$ is the partition function, obtained by summing over all possible hidden and visible vectors

$$\mathscr{Z} = \sum_{\{\mathbf{v},\mathbf{h}\}} e^{-E}. \tag{2.3}$$

As usual, to determine the marginal distribution of a visible vector, sum over the state space of hidden vectors

$$p(\mathbf{v}) = \frac{1}{\mathscr{Z}} \sum_{\{\mathbf{h}\}} e^{-E}. \tag{2.4}$$

Similarly, the marginal distribution of a hidden vector is

$$p(\mathbf{h}) = \frac{1}{\mathscr{Z}} \sum_{\{\mathbf{v}\}} e^{-E}. \tag{2.5}$$

The weights, W_{ia} and biases $b_i^{(v)}$, $b_a^{(h)}$ are determined during training. Training strives to match the model distribution $p(\mathbf{v})$ to the distribution $q(\mathbf{v})$ defined by the data and it achieves this by minimizing the Kullback-Leibler (KL) divergence, which is given by

$$\begin{aligned} D_{KL}(q||p) &= \sum_{\{v\}} q(v) \left(\log\left(q(v)\right) - \log\left(p(v)\right) \right) \\ &= \sum_{\{v\}} q(v) \log\left(\frac{q(v)}{p(v)} \right). \end{aligned} \tag{2.6}$$

The KL divergence is a measure of how much information is lost when approximating the actual distribution with the model distribution [42]. Training adjusts $\{W_{ia}, b_i^{(v)}, b_a^{(h)}\}$ to minimize the KL divergence. Gradients of the KL divergence used to update the parameters of the RBM are computed as follows

$$\frac{\partial D_{KL}(q||p)}{\partial W_{ia}} = \langle v_i h_a \rangle_{data} - \langle v_i h_a \rangle_{model}, \tag{2.7}$$

$$\frac{\partial D_{KL}(q||p)}{\partial b_i^{(v)}} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model}, \tag{2.8}$$

$$\frac{\partial D_{KL}(q||p)}{\partial b_a^{(h)}} = \langle h_a \rangle_{data} - \langle h_a \rangle_{model}, \tag{2.9}$$

where the expectation values appearing above are easily derived using (2.2). They are given explicitly in Appendix A.1. The data set contains an enormous number N_s of samples implying that the method just outlined is numerically intractable: the sum over the whole state space of visible and hidden vectors is too expensive. In this study we consider networks with N_v in the range of 100 to 1024 nodes and N_h in the range of 81 to 256 nodes. The number of samples N_s we sum over lies in the range of 2^{81} and 2^{1024}. To make progress, we approximate the KL divergence by the contrastive divergence [39]. Rather than summing over the entire state space of visible and hidden vectors, one simply sets the states of the visible units to the training data [41]. This is an enormous simplification. Given a set of

visible vectors, the hidden vectors are sampled by setting each h_a to 1 with probability

$$p(h_a{=}\,1|\mathbf{v}) = \frac{1}{2}\left(1+\tanh\left(\sum_i W_{ia}v_i + b_a^{(h)}\right)\right). \tag{2.10}$$

Likewise, given a set of h_a, we are able to sample visible vectors by setting each v_i to 1 with probability

$$p(v_i{=}\,1|\mathbf{h}) = \frac{1}{2}\left(1+\tanh\left(\sum_a W_{ia}h_a + b_i^{v}\right)\right). \tag{2.11}$$

Expectation values for the data are computed using $\hat{\mathbf{h}}$, generated using (2.10) and $\hat{\mathbf{v}}$, provided by the training data set.

To determine model expectation values, determine a sample of visible vectors $\{\tilde{\mathbf{v}}\}$ and a sample of the hidden vectors $\{\tilde{\mathbf{h}}\}$, using the equations (2.11) and (2.10) as we now explain. The set $\{\tilde{\mathbf{v}}\}$, is calculated using $\{\hat{\mathbf{h}}\}$ and equation (2.11). We then determine $\{\tilde{\mathbf{h}}\}$, using $\{\tilde{\mathbf{v}}\}$ and equation (2.10). The equations for the Ath vectors in the sets $\{\hat{\mathbf{h}}\}$, $\{\tilde{\mathbf{v}}\}$ and $\{\tilde{\mathbf{h}}\}$ are thus

$$\hat{h}_a^{(A)} = \tanh\left(\sum_i W_{ia}\hat{v}_i^{(A)} + b_a^{(h)}\right), \tag{2.12}$$

$$\tilde{v}_i^{(A)} = \tanh\left(\sum_a W_{ia}\hat{h}_a^{(A)} + b_i^{(v)}\right), \tag{2.13}$$

$$\tilde{h}_a^{(A)} = \tanh\left(\sum_i W_{ia}\tilde{v}_i^{(A)} + b_a^{(h)}\right), \tag{2.14}$$

Expectation values of the model are approximated using these sets. Again, summing these much smaller sets (and not the complete space of hidden and visible vectors) is an enormous simplification.

Using this approximation the expressions used to train the RBM are

$$\langle v_i h_a\rangle_{data} = \frac{1}{N_s}\sum_A \hat{v}_i^{(A)}\hat{h_a}^{(A)}, \tag{2.15}$$

$$\langle v_i h_a\rangle_{model} = \frac{1}{N_s}\sum_A \tilde{v}_i^{(A)}\tilde{h_a}^{(A)}, \tag{2.16}$$

$$\langle v_i\rangle_{data} = \frac{1}{N_s}\sum_A \hat{v}_i^{(A)}, \tag{2.17}$$

$$\langle v_i \rangle_{model} = \frac{1}{N_s} \sum_A \tilde{v_i}^{(A)}, \tag{2.18}$$

$$\langle h_a \rangle_{data} = \frac{1}{N_s} \sum_A \hat{h_a}^{(A)}, \tag{2.19}$$

$$\langle h_a \rangle_{model} = \frac{1}{N_s} \sum_A \tilde{h_a}^{(A)}, \tag{2.20}$$

where $A = 1, 2, 3, \ldots, N_s$ denotes samples in the training data set made up of N_s samples. These approximations achieve a dramatic speed up in training. Although this method performs well in practice [18, 43, 44], it is difficult to understand when and why the approximations work [41, 45]. This approximation does not follow the gradient of any function [46].

2.2.2 RG

RG is a tool used routinely in quantum field theory and statistical mechanics [30]. RG coarse grains by first organizing the theory according to length scales and then averaging over the short distance degrees of freedom. The result is an effective theory for the long distance degrees of freedom. RG thus gives a systematic procedure to determine the dynamical laws governing macroscopic physics of a system with given microscopic laws, and it achieves this by employing coarse graining. The analogy to deep learning should be evident: deep learning also extracts regularities from massive data sets.

At this point it is helpful to make a comment on what a field is. A field is a type of observable. A very simple example of a field could be the temperature inside a room. The measured value of the temperature depends on exactly where in the room you make the measurement[3] and when you make the measurement[4]. Anything that can be measured everywhere and/or everywhen is an example of a field.

To illustrate RG consider the example provided by quantum field theory. To have a concrete example in mind, which is relevant to the discussion that follows, we might study a field $\phi(x)$ describing the magnetization inside a magnet. The value of the field $\phi(x)$ gives the value of the magnetization at position x. $\phi(x)$ can be manipulated as we would normally manipulate a function of x. In particular, we can take derivatives of $\phi(x)$ with respect to x

[3] For example, there maybe an air conditioner in the room. Points closer to the air conditioner will be cooler.

[4] Temperature measurements at midnight in the middle of winter will typically be lower than measurements at midday in the middle of summer.

and we can take its Fourier transform to obtain $\phi(k)$. Observables $\mathscr{O}$ are functions (usually polynomials) of the field and its derivatives. Examples of observables are the energy or momentum of the field. To calculate the expected value $\langle\mathscr{O}\rangle$ of observable $\mathscr{O}$, integrate (i.e. average) over all possible field configurations

$$\langle\mathscr{O}\rangle = \int [d\phi] e^{-S}\mathscr{O}. \tag{2.21}$$

To make sense of this integral one can work on a lattice. Here we use the term "lattice" to denote an ordered array of points, and we imagine replacing the continuum of space with this discrete structure, so that the set of all possible positions in space is now a discrete set. The integral over all possible field configurations then becomes a product of ordinary integrals, with one integral over the allowed range of the field, at each lattice site. The range of the field is usually taken to be the real number field. The factor e^{-S}, which defines a probability measure on the space of fields, depends on the theory considered. S is called the action of the theory and is also a polynomial in the field and its derivatives, with the coefficients of the polynomial providing the parameters of the theory, things like couplings and masses. A theory is defined by specifying S.

To coarse grain, express the position space field in terms of momentum space components

$$\phi(x) = \int dk e^{ik\cdot x}\phi(k). \tag{2.22}$$

$e^{ik\cdot x}$ oscillates in position space with wavelength $\frac{2\pi}{k}$. High momentum (big k) components have small wavelengths and encode small distance structure. Low momentum components have huge wavelengths and describe large distance structure. Declare there is a smallest possible structure, implemented by cutting off the momentum modes at a large momentum Λ as follows

$$[d\phi] = \prod_{k<\Lambda} d\phi(k). \tag{2.23}$$

RG breaks the integration measure into high and low momentum components $[d\phi] = [d\phi_<][d\phi_>]$ where

$$\begin{aligned} [d\phi_<] &= \prod_{k<(1-\varepsilon)\Lambda} d\phi(k) \\ [d\phi_>] &= \prod_{(1-\varepsilon)\Lambda<k<\Lambda} d\phi(k). \end{aligned} \tag{2.24}$$

The dimensionless parameter ε defines the split between the two sets of components. In the end we imagine taking $\varepsilon \to 0$ as explained below. RG considers observables that depend only on large scale structure of the theory, i.e. observables that depend only on $\phi_<$. In this case, when computing the expected value of $\mathcal{O}$ we can pull $\mathcal{O}$ out of the integral over $\phi_>$ and integrate over the high momentum components

$$\begin{aligned} \langle \mathcal{O}(\phi_<) \rangle &= \int [d\phi_<] \int [d\phi_>]\, e^{-S} \mathcal{O}(\phi_<) \\ &= \int [d\phi_<]\, e^{-S_{\text{eff}}} \mathcal{O}(\phi_<). \end{aligned} \tag{2.25}$$

This procedure of splitting momentum components into two sets and integrating over the large momenta defines a new action S_{eff}. Repeating the procedure many times defines the RG flow under which S_{eff} changes continuously. To obtain a continuous flow we should take the limit $\varepsilon \to 0$, so that the procedure needs to be repeated an infinite number of times to flow to low momentum. The parameter ε should be thought of as a step size in a discrete flow. It is not a physical parameter and must be taken small enough that the results of computations are independent of ε. After the flow, one is left with an integral over the very long wavelength modes. This completes the coarse graining: we have a new theory defined by S_{eff}. The new theory uses only long wavelength components of the field and correctly reproduces the expected value of any observable depending only on long wavelength components. Values of the parameters of the theory, which appear in S_{eff}, change under this transformation. In general, many possible terms are generated and appear in S_{eff}. Each possible term defines a coupling of the theory. Each coupling can be classified as marginal (the size of the coupling is unchanged by the RG flow), relevant (the coupling grows under the flow) or irrelevant (the coupling goes to zero under the flow). It is a dramatic insight of Wilson that almost all couplings in any given quantum field theory are irrelevant and so the low energy theory is characterized by a handful of parameters. This is a dramatic (experimentally verified) simplicity hidden in the rather complicated quantum field theory. This simplicity explains why "simple large scale patterns" can emerge from "complicated short distance data". The possibility that the same simplicity is at work in unsupervised deep learning is a key motivation of this thesis.

Although the equation (2.25) defines the relationship between S and S_{eff}, the connection is rather abstract and a few clarifying remarks are in order. Consider the situation in which we started with an action S and we have flowed to obtain some effective low energy dynamics S_{eff}. The action S is the original action of the theory. In the case of a magnet, this would describe the dynamics of atomic spins, where the relevant length scale is 10^{-10}m. The effective action would describe the dynamics of a classical magnet, where the relevant length

scale may be 10^{-3}m or even larger. The renormalization group is the coarse graining that constructs the classical macroscopic physics from the microscopic physics.

Conceptually, the coarse graining performed by RG is well defined. Computationally, it is almost impossible to carry out. To develop a useful calculation scheme, partition the momentum components into tiny sets (i.e. follow (B.21) with ε infinitesimal) and ask what happens when we average over a single tiny set. Two things happen: couplings g_i change $\delta g_i = \beta_i \varepsilon$ and the strength of the field changes $\delta\phi = \gamma\varepsilon\phi$. One can prove that all observables built using n fields will obey the Callan-Symanzik equation [47]

$$\left(\mu\frac{\partial}{\partial\mu} + \sum_i \beta_i \frac{\partial}{\partial g_i} + n\gamma\right)\langle\mathscr{O}\rangle = 0. \tag{2.26}$$

The parameter μ here defines the scale of the effective theory: the smallest wavelength in the effective theory is $\frac{2\pi}{\mu}$. This equation provides a remarkable and simple description of the RG coarse graining that captures the essential features of the long distance effective theory. In practice correlation functions are computed and then inserted into the Callan-Symanzik equation. The β_i and γ functions are then read from the resulting equations.

If RG (or a variant of it) is relevant to understanding deep learning, it makes concrete suggestions for the resulting theory. For example, is there an analogue of the Callan-Symanzik equation? One might assign beta functions $\beta_{ia}, \beta_i^{(v)}, \beta_a^{(h)}$ to the weights W_{ia} and biases $b_i^{(v)}, b_a^{(h)}$. These would determine which parameters of the RBM are relevant, irrelevant or marginal.

The RG flow halts at a fixed point, described by a conformal field theory. This field theory enjoys additional symmetries including scale invariance. It is interesting to note that the possibility that scale invariance plays a role in unsupervised deep learning has been raised in [29, 9–11, 31].

Although we have focused mainly on a physical model in this article, we should point out that there are many other applications of the renormalization group formalism. For example, RG has been used to understand the spread of forest fires [48], in the modeling of the spread of infectious diseases in epidemiology [49], for the prediction of earthquakes [49] and more generally, to any system with self organized criticality [50].

2.2.3 Ising model

The Ising model is a model for a magnet. The two dimensional model has a discrete variable, called a spin, $\sigma_{\vec{k}} = \pm 1$ on each site of a rectangular lattice. The sites are labeled by a two dimensional vector $\vec{k}$, which has integer components. A state of the system is given by

specifying a collection of spins, $\{\sigma_{\vec{k}}\}$, one for each site in the lattice. To refer to a specific state of the spin system we use the notation $\sigma = \{\sigma_{\vec{k}}\}$. Spins on adjacent sites i and j interact with strength J_{ij}. Each spin will also interact with an external magnetic field h_j, with strength μ. The energy of a given configuration $\{\sigma_{\vec{k}}\}$ of spins is determined by the Hamiltonian

$$H = -\sum_{\langle i\,j\rangle} J_{ij}\sigma_i\sigma_j - \mu\sum_j h_j\sigma_j, \tag{2.27}$$

where the first sum is over adjacent pairs, indicated by $\langle i\,j\rangle$. We simplify the model by setting the external field to zero $h_j = 0$, and by choosing the couplings in the most symmetric possible way $J_{ij} = J$. Since J is an energy we can set it to 1 by choosing units appropriately. The probability of configuration $\sigma = \{\sigma_{\vec{k}}\}$ of spins is given by the Boltzmann distribution, with inverse temperature $\beta \geq 0$

$$P_\beta(\sigma) = \frac{e^{-\beta H(\{\sigma_{\vec{k}}\})}}{Z_\beta}, \tag{2.28}$$

where the constant Z_β, the partition function, is given by

$$Z_\beta = \sum_{\{\sigma_{\vec{k}}\}} e^{-\beta H(\sigma)}. \tag{2.29}$$

Averages of physical observables are defined by

$$\langle f\rangle_\beta = \sum_\sigma f(\sigma)P_\beta(\sigma). \tag{2.30}$$

We study unsupervised learning of the Ising model by an RBM. The visible data that is used to train the network is generated using the probability measure (2.28). The lattices have a total of $L_v \times L_v$ sites, with each site indexed by a position vector $\vec{k}$. We rearrange this array of spins σ into an $N_v = L_v \times L_v$ dimensional vector by concatenating the rows of the given array. These components of these vectors are the training data input to the visible nodes of the RBM.

There are good reasons to focus on the Ising model. The model has a fixed point in its RG flow. The fixed point is described by a well known conformal field theory (CFT) [51]. This fixed point is an unstable fixed point meaning that generic flows move away from the fixed point. We must tune things carefully if we are to terminate on the fixed point. This tuning is necessary because there is a relevant operator present in the spectrum of the conformal field theory and it tends to push us away from the fixed point. We need to tune the temperature. If the temperature is slightly above the critical temperature, thermal fluctuations

destroy the long range correlations that are forming, whilst if we are slightly below the critical temperature, the tendency of spins to align dominates and we find a state with all spins aligned and no fluctuations at all. It is only exactly at the critical temperature that the system exhibits the interesting long range correlations that are described by the CFT. The papers [10, 11] argue that the RBM flow always flows to the fixed point. This challenges conventional wisdom and it suggests a different kind of coarse graining to that employed by RG, is at work. A distinct proposal [9] claims that the RG flow arises by stacking RBMs to produce a classic deep learning scenario. Each layer of the deep network performs a step in the flow.

At the Ising model fixed point, detailed checks of both proposals are possible. There are CFT observables, known as primary operators, whose correlators are power laws of distances on the lattice. The powers entering these power laws are known, so that we have a rich and detailed data set that the RBM must reproduce if it is indeed performing an RG coarse graining. This is a compelling motivation for the model. Another advantage of the model is simplicity: it is a model of spins which take the values ± 1 so it defines a simple model with discrete variables, well suited to numerical study and naturally accommodated in the RBM framework. Finally, the Ising model is not that far removed from real world applications: the Ising Hamiltonian favors configurations with aligned neighboring spins. Thus, at low enough temperatures "smooth" slowly varying configurations of spins are favored. This is similar to data defining images for example, where neighboring pixels are likely to have the same color. In slightly poetic language one could say that at low temperatures the Ising model favors pictures and not speckle.

We end this section with a summary of the most relevant features of the Ising model fixed point. At the critical temperature

$$T_c = \frac{2J}{k \ln(1+\sqrt{2})}, \tag{2.31}$$

where J is the interaction strength and k is the Boltzmann constant, the Ising model undergoes a second order phase transition. The critical temperature is given by $T_c \approx 2.269$ when $J = 1$. There are two competing phases: an ordered (low temperature) phase in which spins align producing a macroscopic magnetization, and a disordered (high temperature) phase in which spins fluctuate randomly and the magnetization averages to zero. At the critical point the Ising model develops a full conformal invariance and one can use the full power of conformal symmetry to tackle the problem. The field which takes values ± 1 in the Ising model is a primary field, of dimension $\Delta = \frac{1}{8}$. The two and three point correlation functions of primary

fields are determined by conformal invariance to be

$$\langle \phi(\vec{x}_1)\phi(\vec{x}_2)\rangle = \frac{B_1}{|\vec{x}_1-\vec{x}_2|^{2\Delta}}, \tag{2.32}$$

$$\langle \phi(\vec{x}_1)\phi(\vec{x}_2)\phi(\vec{x}_3)\rangle = \frac{B_2}{|\vec{x}_1-\vec{x}_2|^{\Delta}|\vec{x}_1-\vec{x}_3|^{\Delta}|\vec{x}_2-\vec{x}_3|^{\Delta}}, \tag{2.33}$$

where B_1 and B_2 are constants. Since we study the Ising model on a lattice, the positions $\vec{x}_1$, $\vec{x}_2$ and $\vec{x}_3$ appearing in the above correlation functions refer to sites in a lattice. There is also a primary operator in the Ising model (which we describe below) with a dimension $\Delta = 1$. These correlation functions must be reproduced by the RBM if it is indeed flowing to the critical point of the Ising model.

2.3 Flows derived from learned weights

In this section we consider the RBM flows introduced in [10, 11]. These flows use the weight matrix W_{ia}, and bias vectors $b_i^{(v)}$ and $b_a^{(h)}$, obtained by training, to define a continuous flow from an initial spin configuration to a final spin configuration. The flow appears to exhibit a fascinating behavior: given any initial snapshot, the RBM flows towards the critical point of the Ising model. This is in contrast to the RG which flows away from the fixed point. In addition, the number of spins in the configuration is a constant along the RBM flow. In contrast to this, the number of spins in the configuration decreases along the RG flow, as high energy modes are averaged over to produce the coarse grained description. Despite these differences, the flow of [10, 11] appears to produce configurations ever closer to the critical temperature and these configurations yield impressively accurate predictions for the critical exponents of the Ising magnet. Our goal in this section is to further test if the RBM flow produces configurations at the critical point of the Ising model. We explore the spatial dependence of spin correlations in configurations produced by the flow. Our results prove that on large scales the Ising critical point configurations are correctly reproduced. However, we are also able to prove that as one starts to probe smaller scales there are definite quantifiable departures from the Ising predictions.

2.3.1 RBM flow

RBM flows [10, 11] are generated using equation (2.13) together with the trained weight matrix, W_{ia}, and bias vectors, $b_i^{(v)}$ and $b_a^{(h)}$. Our data set $\hat{v}^{(A)}$ is labeled by an index A. For

each value of A, $\hat{v}^{(A)}$ is a collection of spin values, one for each lattice site. The RBM flow is generated through a series of discrete steps, with each step producing a new data set of the same size as the original. Denote the data set produced after k steps of flow by $\tilde{v}^{(A,k)}$, and by convention we identify $\tilde{v}^{(A,0)}$ with the original training set. Apply equation (2.13) to $\hat{v}^{(A,k)}$ and then apply (2.14) to carry out a single step of the RBM flow, with the result $\tilde{v}^{(A,k+1)}$. The flow proceeds by repeatedly applying equations (2.13) and (2.14). Concretely, for a flow of length n, we have

$$\begin{aligned}
\tilde{v}_i^{(A,1)} &= \tanh\left(\sum_a W_{ia}\hat{h}_a^{(A)} + b_i^{(v)}\right), \\
\tilde{v}_i^{(A,2)} &= \tanh\left(\sum_a W_{ia}\tilde{h}_a^{(A,1)} + b_i^{(v)}\right), \\
&\vdots \\
\tilde{v}_i^{(A,n)} &= \tanh\left(\sum_a W_{ia}\tilde{h}_a^{(A,n-1)} + b_i^{(v)}\right).
\end{aligned} \tag{2.34}$$

where

$$\begin{aligned}
\tilde{h}_a^{(A)} &= \tanh\left(\sum_i W_{ia}\tilde{v}_i^{(A)} + b_a^{(h)}\right), \\
\tilde{h}_a^{(A,1)} &= \tanh\left(\sum_i W_{ia}\tilde{v}_i^{(A,1)} + b_a^{(h)}\right), \\
\tilde{h}_a^{(A,2)} &= \tanh\left(\sum_i W_{ia}\tilde{v}_i^{(A,2)} + b_a^{(h)}\right), \\
&\vdots \\
\tilde{h}_a^{(A,n)} &= \tanh\left(\sum_i W_{ia}\tilde{v}_i^{(A,n)} + b_a^{(h)}\right).
\end{aligned} \tag{2.35}$$

Note that the length of the vector $\tilde{v}^{(A,k)}$ is a constant of the flow and consequently there is not obviously any coarse graining implemented.

2.3.2 Numerical results

This section considers statistical properties of configurations produced by the RBM flow. At the Ising critical point, the theory enjoys a conformal invariance. Using this symmetry a special class of operators with a definite scaling dimension Δ can be identified. The utility of these operators is that their spatial two-point correlation functions drop off as a known power of the distance between the two operators, as reviewed above in equation (2.32). These two-point functions can be evaluated using the RBM flow configurations and, if these configurations are critical Ising states, they must reproduce the known correlation functions. This is one of the checks performed and it detects discrepancies with the Ising model predictions. There are two primary operators we consider. This first is the basic spin variable minus its average value $s_{ij} = \sigma_{ij} - \bar{\sigma}$. The prediction for the two-point function is (2.32) with $\Delta_s = \frac{1}{8}$. This correlator falls off rather slowly, so that this two-point function probes the large scale features of the RBM flow configurations. The RBM flow nicely reproduces this correlator and in fact, this is enough to reproduce the critical exponent for the Ising model consistent with the results of [11]. One should note however that our computation and those of [11] could very well have disagreed, since they probe different things. The critical exponent evaluated in [11] uses the magnetization computed from different flows generated by the RBM, at temperatures around the critical temperature. Magnetization measures the average of the spin in the lattice. It is blind to the spatial location of each spin. On the other hand, the two-point correlation function is entirely determined by the spatial location of spins in a single flow configuration. Thus, the two-point correlation function uses data at a single temperature, but uses detailed spatial dependence of the lattice state. We also consider a second primary operator

$$\varepsilon_{ij} = s_{ij} \cdot (s_{i+1,j} + s_{i-1,j} + s_{i,j+1} + s_{i,j-1}) - \bar{\varepsilon}, \tag{2.36}$$

which has $\Delta_\varepsilon = 1$. We have again subtracted off the average value of the operator. This correlation function falls off much faster and is consequently a probe of shorter scale features of the RBM configurations. The RBM flow fails to reproduce this correlation function, indicating that the RBM configurations differ from those of the critical Ising model. They have the same long distance features, but differ on shorter length scales.

Consider an RBM network with 100 visible nodes and 81 hidden nodes. This corresponds to an input lattice of size 10×10 and an output lattice of size 9×9. The number of visible and hidden nodes is chosen to match [11], so that we can compare our results to existing literature.

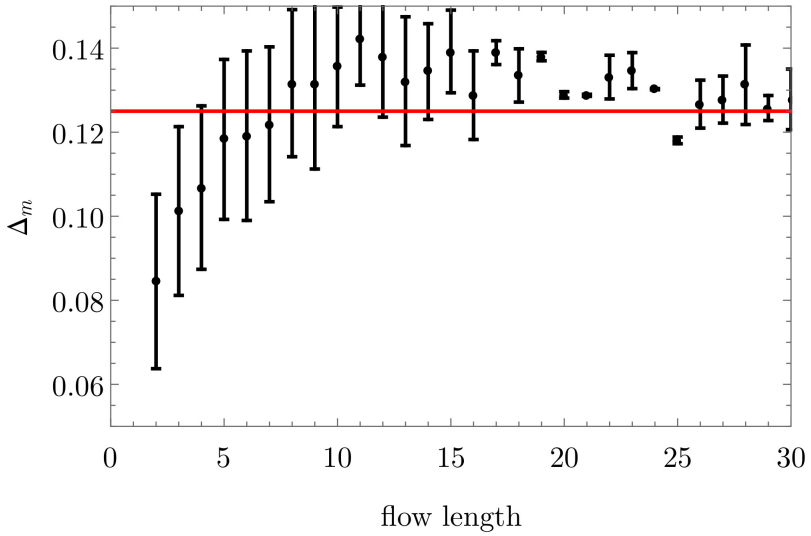


Fig. 2.2 Estimates of the scaling dimension Δ_m versus flow length, obtained using the average magnetization of flows at temperatures $T = 2.1$, 2.2 and 2.3. The red line indicates the value of $\Delta = 0.125$ at the critical point. After approximately 8 flows, Δ_m converges to this critical value. The error bars are determined using Mathematica's NonlinearModelFit function. Mathematica uses the Student's t-distribution to calculate a confidence interval for the given parameters with a 90% confidence level.

We would like to study a lattice that is as small as possible but large enough to detect the power law fall off of the correlation functions we study. The power law behavior is given in the scaling dimensions Δ_s and Δ_ε. We demonstrate that when studying a lattice of size 9×9 or larger we correctly determine the values for Δ_s and Δ_ε using configurations generated from MC simulations. These results are shown in Figures 2.6a and 2.5b.

The network trains on data generated by Monte Carlo simulations which use the Boltzmann distribution given in equation (2.28) [52]. The training data set includes 20000 samples at each temperature, ranging from 0 to 5.9 in increments of 0.1. This gives a total of 1200000 configurations. Training uses 10000 iterations of contrastive divergence, performed with the update equations (2.12) to (2.20) which are derived from equations (2.7), (2.8) and (2.9) [12].

Once the flow configurations are generated, following [10, 11], a supervised network is used to measure the temperature of each flow.

The supervised network allows us to measure discrete temperatures of $T = 0, 0.1, \ldots, 5.9$.

We train a network which consists of three layers, an input layer with 100 nodes, a hidden layer with 80 nodes and an output layer with 60 nodes which correspond to the 60 temperatures we want to measure. All nodes in the input layer are connected to all nodes in

the hidden layer, and all nodes in the hidden layer are connected to all nodes in the output layer. No connections between nodes within the same layer exist.

The Ath sample of input data, $Z_A^{(1)}$ is transformed by the hidden layer using the hyperbolic tangent function $f(x) = \tanh(x)$ as follows

$$Z_{Aa}^{(2)} = f(\sum_i Z_{Ai}^{(1)} W_{ia}^{(1)} + b_a^{(1)}). \tag{2.37}$$

The output layer then transforms $Z_A^{(2)}$ into an output probability using the softmax function $g(x)$

$$Z_{A\mu}^{(3)} = g(\sum_a Z_{Aa}^{(2)} W_{a\mu}^{(2)} + b_\mu^{(2)}) =: g(U_{A\mu}^{(2)}), \tag{2.38}$$

where the softmax function normalizes the output to sum to 1 as follows

$$g(U_{A\mu}^{(2)}) = \frac{\exp(U_{A\mu}^{(2)})}{\sum_\nu \exp(U_{A\nu}^{(2)})}. \tag{2.39}$$

The output of the trained network can thus be interpreted as probabilities for the temperatures we measure. The highest probability in the output is taken as the temperature for the given input.

We make use of the Keras library [53] to train this network using the back-propagation algorithm [54]. The cost function used to train the network is the KL divergence as is used in [11]. We choose a learning rate of $\varepsilon = 0.1$ and train the network for 3000 epochs. In Figure 2.3 we show the validation and training loss versus the number of training epochs. The supervised network is trained using the same data set used to train the RBM as well as corresponding labels for each vector. These labels are one hot encoded vectors which correspond to the temperatures $T = 0, 0.1, \ldots, 5.9$. Here we use a split of 40 % of the input data for validation and 60 % of the input data for training. Figure 2.3 shows that the supervised network has converged to a loss very close to zero after 3000 epochs.

To estimate Δ using magnetization the study [11] selects flows at temperatures close to T_c, where the average magnetization m depends on temperature as

$$m \approx \frac{A|T - T_c|^{\Delta_m}}{T_c}. \tag{2.40}$$

In [11] the magnetization is expanded about the critical point to give values for $A = 1.22$ and $\Delta_m = 1/8 = 0.125$

$$m \approx 1.222 \frac{|T_c - T|^{1/8}}{T_c}. \tag{2.41}$$

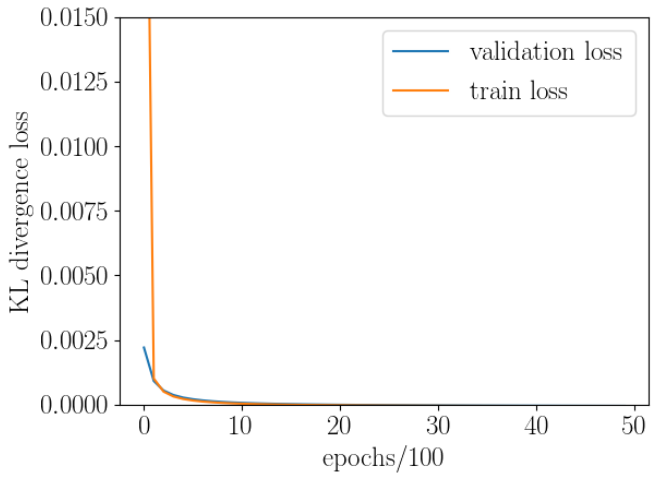


Fig. 2.3 Plot showing the KL divergence loss function of the supervised temperature measuring network versus the number of epochs divided by 100. This network consists of an input layer with 100 nodes, a hidden layer with 80 nodes and and output layer of 60 nodes.

We denote Δ obtained by this fitting as Δ_m. The fit also determines A. The value of $T_c = 2.269$ is a known theoretical value for the 2D Ising model with coupling strength $J = 1$. The fit uses the magnetization computed at temperatures $T = 2.1$, 2.2 and 2.3. A plot of Δ_m versus flow length is given in Figure 2.2. The error bars shown in Figure 2.2 (and all subsequent plots) are determined using Mathematica's NonlinearModelFit function. The error bars show the standard error obtained from the regression. Mathematica uses the Student's t-distribution to calculate a confidence interval for the given parameters with a 90% confidence level.

Our results indicate that we converge to the correct critical value $\Delta_m = 0.125$ for flows of length 26. It is evident from Figure 2.2 that the flow converges to the theoretical value depicted by the red horizontal line. The convergence of the flow is also reflected as a decrease in the size of the error bars as the flow proceeds. In [11] the value for A/T_c is found to be 0.931. The value of A/T_c which we find after convergence is 0.942 with a 90% confidence interval within ± 0.0168794 of this value. Although the values we obtain for A and Δ_m are consistent with the results of [11], they have used a flow of length 9, at which point Δ_m is correctly determined. As just described, we need longer flows for convergence. The fitting of Δ_m and A, for a flow length of 26, is shown in Figure 2.4.

We now shift our focus to consider spatial two-point correlation functions computed using the configurations generated by the RBM flows. The correlators are calculated using the flow configuration and the result is then fitted to the function in equation (2.32) to estimate Δ. We denote this estimate by Δ_s as it is determined using spatial information. For RBM

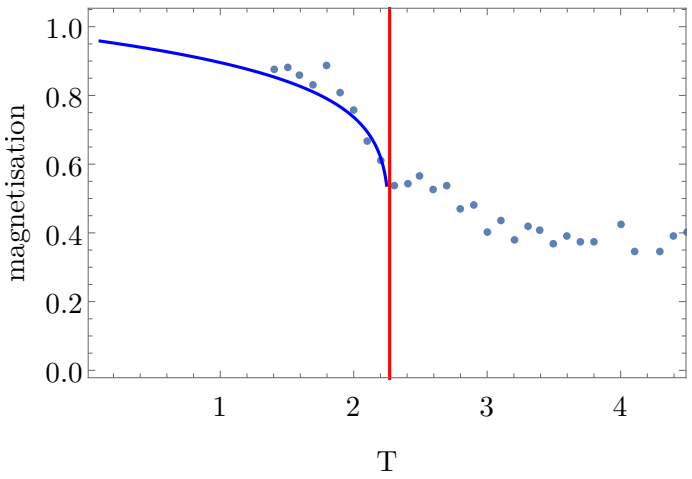
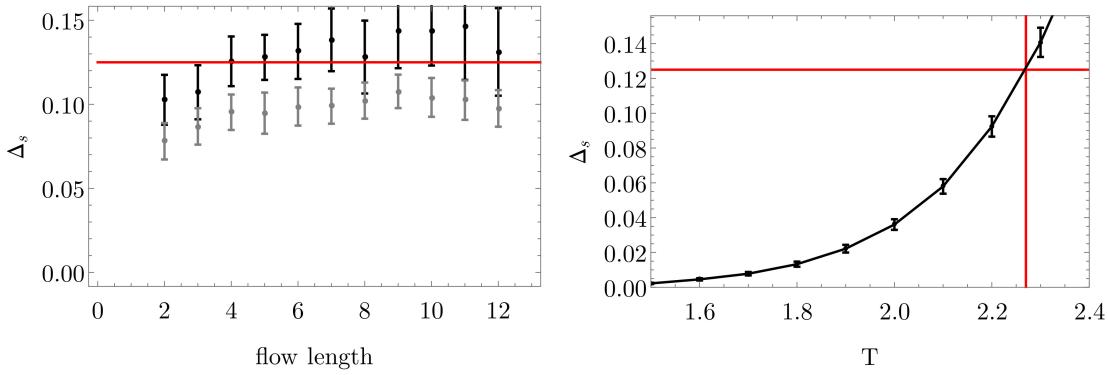


Fig. 2.4 Plot showing the fit for equation (2.40). The blue line shows the function $m = 0.942(T - 2.269)^{0.126}$. This function is fitted using the dots which show the average magnetization for flows of length 26 at various temperatures measured using the supervised network. The vertical red line shows the critical temperature of $T_c \approx 2.269$. Equation (2.40) is fitted using data points for temperatures near T_c.

flows at the critical temperature, the prediction which is determined by theory is $\Delta_s = 0.125$ at $T = 2.269$, as explained above.

We expect that for temperatures below the critical temperature Δ_s will be less than the theoretical value of 0.125. For temperatures that are above the critical temperature we expect that Δ_s will be greater than 0.125. With a lattice of 10 by 10 spins, we find $\Delta_s = 0.1263$ using Monte Carlo Ising model configurations. A plot showing this estimate can be seen in Figure 2.5b. The point of this exercise is to demonstrate that a lattice of size 10 by 10 is large enough to estimate the scaling dimension of interest and to verify the integrity of the data set used in the numerical simulations.

Figure 2.5a shows the scaling dimension Δ_s versus the flow length, for RBM flows at temperatures of 2.2 (in gray) and 2.3 (in black). The red horizontal line indicates the scaling dimension at the critical point. The results are intuitively appealing. The gray points in Figure 2.5a show estimates of Δ_s from flows slightly below the critical temperature, where the scaling dimension is slightly underestimated. Below the critical temperature spins are more likely to align and so the correlator should fall off more slowly than at the critical temperature. This is what our results show. The black points in Figure 2.5a show Δ_s estimated using flows slightly above the critical temperature. The scaling dimension is over estimated, again as expected. Selecting flows at T_c would determine the scaling dimension in between the values shown in Figure 2.5a. This gives a value very close to the theoretical value of $\Delta_s = 0.125$.



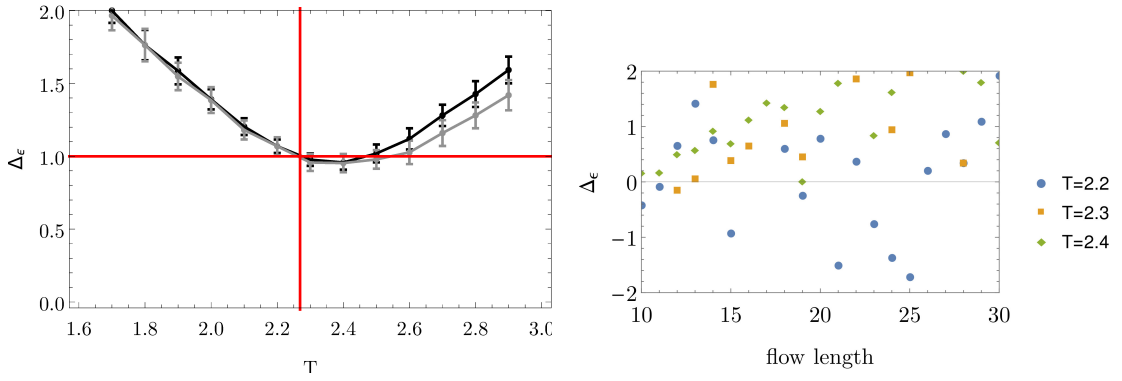
(a) (b)

Fig. 2.5 Plot showing the estimated scaling dimension Δ_s versus flow length using the two-point correlation function for (a) flows at $T = 2.2$ (in gray) and flows at $T = 2.3$ (in black). Plot (b) shows the estimated scaling dimension versus temperature for the Ising model data used for training. The error bars are determined using Mathematica's NonlinearModelFit function. Mathematica uses the Student's t-distribution to calculate a confidence interval for the given parameters with a 90% confidence level.

The two-point correlation functions for the spin variable establish that the critical Ising states and the states produced by the RBM flow share the same large scale spatial features. We will now consider the two-point correlation function of the ε_{ij} field, which probes spatial features on a smaller scale. Using critical Ising data generated using Monte Carlo, on a lattice of size 10 by 10 and 9 by 9, we estimate Δ_ε at various temperatures as shown in Figure 2.6a. We can estimate the error for the point we are interested in, at $\Delta_\varepsilon = 1$ and $T = 2.269$ which is depicted by the red vertical and horizontal lines shown in Figure 2.6a. The error bar of the estimate for the grey line (9×9 lattice) is ± 0.059 for an estimated value of Δ_ε being 1.013. For the black line (10×10 lattice) we have an error bar of ± 0.044 on an estimated value of 1.023 for Δ_ε. This is determined using the average of the values as well as the errors on either side of the red vertical line at $T = 2.269$.

Figure 2.6a, demonstrates that a lattice of size 9×9 is large enough to correctly determine the ε scaling dimension Δ_ε.

The intersection of the red horizontal and vertical lines cross the critical temperature and prediction $\Delta_\varepsilon = 1$. Interpolating the Ising data with a continuous curve, we would pass through the intersection point, as predicted. These numerical results again demonstrate that a lattice of size 10 by 10 is large enough for the questions we consider. The RBM flows are unable to confirm this prediction. Indeed, the RBM flows near T_c are summarized in Figure 2.6b. None of the three temperatures shown have a value of Δ_ε that converges with flow length.



(a) (b)

Fig. 2.6 Plots showing Δ_ε calculated using (a) Monte Carlo Ising model data on a 10 by 10 lattice (in black) and a 9 by 9 lattice (grey), (b) RBM flows at a temperature of T =2.2, 2.3 and 2.4. Error bars in plot (a) indicate a 90% confidence interval. No error bars are shown in (b) as the error bars are larger than the y range. The error bars shown in (a) are determined using Mathematica's NonlinearModelFit function. Mathematica uses the Student's t-distribution to calculate a confidence interval for the given parameters with a 90% confidence level.

The fact that the RBM produces configurations that correctly reproduce the correlation function of the spin field s_{ij} but not of the ε_{ij} implies that although the spatial correlations encoded into the RBM flow configurations agree with those of the critical Ising configurations at long length scales, the two start to differ on smaller length scales. This conclusion agrees with [11] which also finds differences between the RBM flow and RG. [11] considers $h \neq 0$ and uses different arguments to reach the conclusion.

2.4 Flows derived from deep learning

The RBM flows of the previous section provide one possible link to RG. An independent line reasoning, developed in [9], claims a mapping between unsupervised deep learning and RG. The idea is not that there is an analogy between deep learning and RG, but rather, that the two are to be identified. The argument for this identity starts from the energy function of the RBM, which is

$$E(\{v_i, h_a\}) = b_a h_a + v_i W_{ia} h_a + c_i v_i. \tag{2.42}$$

This energy determines the probability of obtaining configuration $\{v_i, h_a\}$ as

$$p_\lambda(\{v_i, h_a\}) = \frac{e^{-E(\{v_i, h_a\})}}{\mathscr{Z}}, \tag{2.43}$$

where $\lambda = \{b_a, W_{ia}, c_i\}$ are the parameters of the RBM model which are tuned during training. Marginal distributions for hidden and visible spins are defined as follows

$$\begin{aligned} p_\lambda(\{h_a\}) &= \sum_{\{v_i\}} p_\lambda(\{v_i, h_a\}) = \mathrm{tr}_{v_i}\, p_\lambda(\{v_i, h_a\}), \\ p_\lambda(\{v_i\}) &= \sum_{\{h_a\}} p_\lambda(\{v_i, h_a\}) = \mathrm{tr}_{h_a}\, p_\lambda(\{v_i, h_a\}). \end{aligned} \tag{2.44}$$

The equations (2.44) are key equations of the RBM. [9] essentially uses these to characterize the RBM. The comparison to RG is made by employing a version of RG known as variational RG. This is an approximate method that can be used to perform the renormalization group transformation in practice. As explained in Appendix A.2, the variational RG uses an operator $T(\{v_i, h_a\})$ defined as follows

$$\frac{e^{H^{RG}_\lambda(\{h_a\})}}{\mathscr{Z}} = \mathrm{tr}_{v_i} \frac{e^{T(\{v_i,h_a\}) - H(\{v_i\})}}{\mathscr{Z}}. \tag{2.45}$$

In this formula, $H(\{v_i\})$ is the microscopic Hamiltonian describing the dynamics of the visible spins and $H^{RG}_\lambda(\{h_a\})$ is the coarse grained Hamiltonian describing the hidden spins where here λ defines the parameters of the variational RG. Block spin averaging is discussed in more detail in Appendix A.2.2. The operator $T(\{v_i, h_a\})$ is required to obey (see in equation (A.12))

$$\mathrm{tr}_{h_a}\, e^{T(\{v_i,h_a\})} = 1, \tag{2.46}$$

which obviously implies that

$$\mathrm{tr}_{h_a}\, e^{T(\{v_i,h_a\}) - H(\{v_i\})} = e^{-H(\{v_i\})}. \tag{2.47}$$

Notice that (2.45) and (2.47) exactly match (2.44) as long as we identify

$$T(\{v_i, h_a\}) = -E(\{v_i, h_a\}) + H(\{v_i\}). \tag{2.48}$$

This then implies that

$$\begin{aligned} \frac{e^{H^{RG}_\lambda(\{h_a\})}}{\mathscr{Z}} &= \mathrm{tr}_{v_i} \frac{e^{T(\{v_i,h_a\}) - H(\{v_i\})}}{\mathscr{Z}} \\ &= \mathrm{tr}_{v_i} \frac{e^{-E(\{v_i,h_a\})}}{\mathscr{Z}} \\ &= \frac{e^{-H^{RBM}_\lambda(\{h_a\})}}{\mathscr{Z}}, \end{aligned} \tag{2.49}$$

which is the central claim of [9].

The above argument proves an equivalence between unsupervised deep learning and RG if and only if the equations (2.44) provide a unique characterization of the joint probability function $p_\lambda(\{v_i,h_a\})$. This is not the case: it is easy to construct functions $p_\lambda(\{v_i,h_a\})$ that obey (2.44), but are nothing like either the RBM or RG joint probability functions. As an example, define

$$\begin{aligned}\rho(\{v_i\}) &= \frac{\mathrm{tr}_{h_a}\left(e^{T(\{v_i,h_a\})-H(\{v_i\})}\right)}{\mathscr{Z}},\\ \tilde{\rho}(\{h_a\}) &= \frac{\mathrm{tr}_{v_i}\left(e^{T(\{v_i,h_a\})-H(\{v_i\})}\right)}{\mathscr{Z}},\end{aligned} \tag{2.50}$$

where

$$\mathscr{Z} = \sum_{v_i,h_a} e^{T(\{v_i,h_a\})-H(\{v_i\})}. \tag{2.51}$$

We clearly have $\mathrm{tr}_{v_i}(\rho(\{v_i\})) = 1 = \mathrm{tr}_{h_a}(\tilde{\rho}(\{h_a\}))$ which implies that

$$A_\lambda(\{v_i,h_a\}) = \tilde{\rho}(\{h_a\})\rho(\{v_i\}), \tag{2.52}$$

obeys (2.44). It is quite clear that in $A_\lambda(\{v_i,h_a\})$ there are no correlations between the hidden and visible spins

$$\begin{aligned}\langle v_j h_b\rangle =& \mathrm{tr}_{v_i,h_a}(v_j h_b A_\lambda(\{v_i,h_a\}))\\ =& \mathrm{tr}_{v_i}(\rho(\{v_i\})v_j)\,\mathrm{tr}_{h_a}(\tilde{\rho}(\{h_a\})h_b)\\ =& \langle v_j\rangle\langle h_b\rangle,\end{aligned} \tag{2.53}$$

so that we would reject it as a possible model of either the RG quantity

$$\mathscr{Z}^{-1} e^{T(\{v_i,h_a\})-H(\{v_i\})}, \tag{2.54}$$

or of the RBM quantity

$$\mathscr{Z}^{-1} e^{-E(\{v_i,h_a\})}. \tag{2.55}$$

In addition to clarifying aspects of the argument of [9], the joint correlation functions between visible and hidden spins can be used to characterize the RG flow, as we now explain. The RG flow "coarse grains" in position space: a "block of spins" is replaced by an effective spin, whose magnitude is the average of the spins it replaces. Since correlations between microscopic spins fall off with distance, an RG coarse graining implies that because the

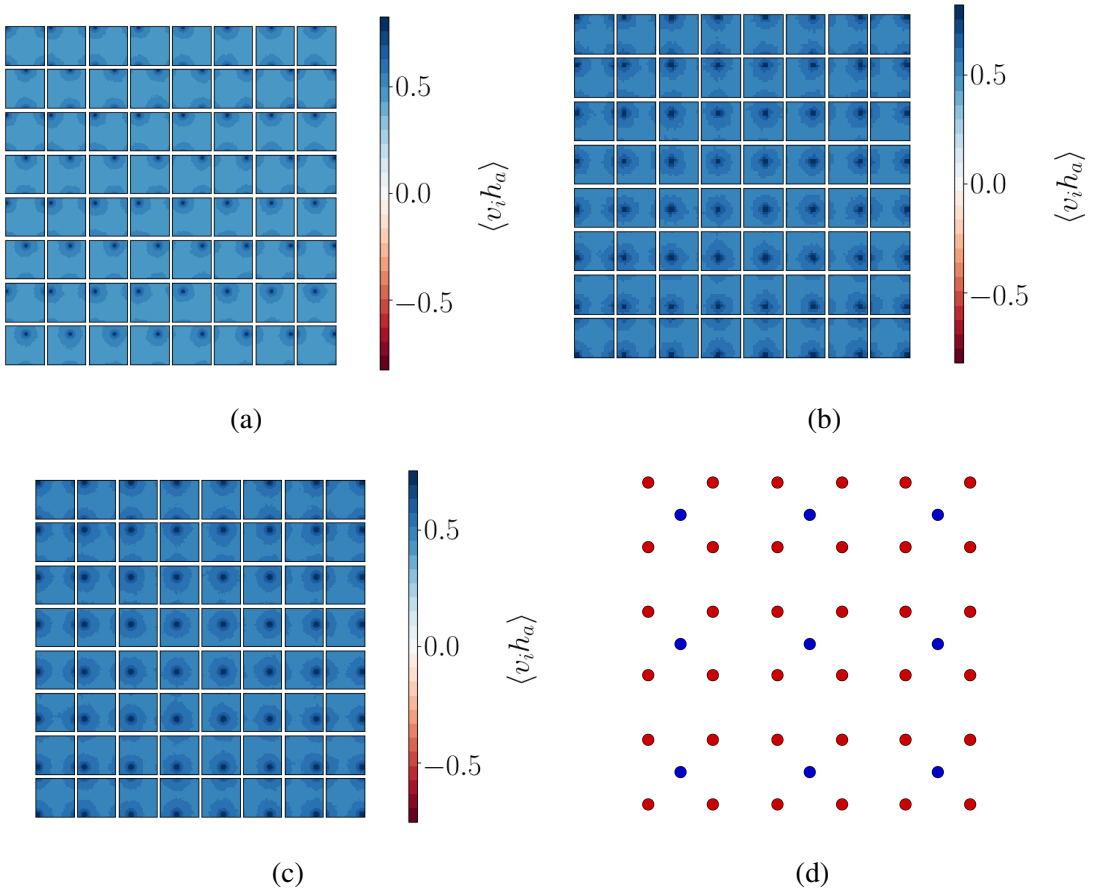


Fig. 2.7 Correlation plots for Ising model visible data with lattice size 32 by 32 at T_c and RG decimated Ising data of sizes 16 by 16 (one step of RG) and 8 by 8 (two steps of RG). (a) shows visible Ising data correlated with configurations resulting after one step of RG. (b) shows correlations between configurations resulting after one step of RG and configurations resulting after two steps of RG. (c) shows correlations between Ising model visible data and configurations resulting after two steps of RG. (d) shows one step of RG. The red dots show the original visible lattice and the blue dots show the lattice obtained after one step of RG. Each blue dot is surrounded by four red dots. The value of the blue dot is determined by averaging the surrounding four red dots.

hidden spin is a linear combination of nearby visible spins, the correlation function between hidden and visible spins reflects a correlation between a hidden spin and a cluster of visible spins. This produces distinctive correlation functions, some examples of which are plotted in Figure 2.7. We will search for this distinctive signal in the $\langle v_i h_a \rangle$ correlator, to find quantitative evidence that unsupervised deep learning is indeed performing an RG coarse graining.

2.4.1 Numerical results

Our numerical study aims to do two things: First, we establish whether there are RG-like patterns present within the correlator $\langle v_i h_a \rangle$, for correlators computed using the patterns generated by an RBM flow. If these patterns are indeed present, this constitutes strong evidence in favor of the connection between RG and deep learning.

The $\langle v_i h_a \rangle$ correlator is calculated using

$$\langle v_i h_a \rangle = \frac{1}{N_s} \sum_{A=1}^{N_s} v_i^{(A)} h_a^{(A)}, \tag{2.56}$$

where $A = 1, 3, \ldots, N_s$ with N_s being the number of samples, $i = 1, 2, \ldots, N_v$ labels the visible nodes within a visible vector and $a = 1, 2, 3, \ldots, N_h$ labels the nodes within a hidden vector.

For each hidden node h_a we can produce a plot which shows how this hidden node is correlated to the $i = 1, 2, 3, \ldots, N_v$ visible nodes, v_i. This gives us a total of N_h plots. Each panel within the plot for h_a shows the N_v correlation values for $\langle v_i h_a \rangle$. By arranging these panels according to the lattice sites of the visible spins we get a grid of $L_v \times L_v = N_v$ values for the correlators $\langle v_i h_a \rangle$, where a is fixed for the given plot and i runs from 1 to N_v.

By doing this we can determine if a given hidden node is correlated to a local patch of visible nodes which are neighbors on the original lattices produced from MC. This local information is not encoded inherently in the RBM so learning about the nearest neighbour interactions present in the 2D Ising model would show promise that RBMs are performing a coarse graining related to that of RG. We find that RG-like patterns do indeed emerge.

Second, according to the proposal of [9], in a deep network each layer that is stacked to produce the depth of the network performs one step in the RG flow. With this interpretation in mind, it may be useful to compare how a network with multiple stacked RBMs learns as compared to a network with a single layer. This issue is explored below.

The training data is a set of 30000 configurations of Ising model 32 by 32 lattices, near the critical temperature $T = 2.269$. The dataset is generated using Monte Carlo simulations. An input lattice length of 32 allows a large enough final configuration even after two steps of RG, corresponding to stacking two RBMs. In each step of the RG, the number of lattice sites is reduced by a factor of 4. Thus, we flow from lattices with 1024 sites to lattices with 64 sites. We enforce periodic boundary conditions. To find signals of RG in the correlation functions the maximum distance between operators in a correlator must be large enough that the spin-spin correlation has dropped to zero. We have confirmed that our lattice is large enough, judged by this criterion.

Having described the conditions of our numerical experiment, we consider the correlators $\langle v_i h_a \rangle$ generated when the hidden neurons h_a are generated from the visible neurons v_i using RG. Our goal is to understand the patterns appearing in correlation functions, that are a signature of the RG. In Figure 2.7d the process of decimation used in our RG is explained. The red dots, representing the visible lattice, are averaged (coarse grained) to produce the blue dots which define the lattice after a single step of the RG. The four spin values located at the red dots surrounding each blue dot are averaged to obtain the value of the spin at the new (blue) lattice point. This process clearly reduces the number of lattice sites by a factor of four.

Using the visible data which populates a 32 by 32 lattice, we populate lattices of size 16 by 16 and 8 by 8 spins by applying the RG and then calculate the various possible $\langle v_i h_a \rangle$ correlations.

Figure 2.7a shows the $\langle v_i h_a \rangle$ correlation function that results from a single RG step. Each panel of the three Figures 2.7a, 2.7b and 2.7c, shows how a given hidden spin is correlated with the visible spins. We can clearly see a peak in correlation values around the spatial location of the hidden spin. This is the signal of RG coarse graining: small spatially localized collections of spins are replaced by their average value. We can go into a little more detail: the patches of large correlation in Figures 2.7b and 2.7c are larger in size than those of Figure 2.7a. This makes sense since each step of the RG implies ever larger spatial regions of the spins are being averaged to produce the coarse grained variables. The fact that the spins that are averaged are spatially localized is a direct consequence of the fact that the Ising model Hamiltonian is local in space so that spatially adjacent spins have similar behaviors. In more general big data settings it may be harder to decide if the coarse graining is RG-like or not, since it might not be clear what is meant by spatial locality.

Having established the signal characteristic of the RG flow, we will now search for this signal in the $\langle v_i h_a \rangle$ correlators computed using the configurations generated from the RBM flow. We consider configurations generated by a stacked network with an RBM having 1024 visible nodes and 256 hidden nodes cascading into a second RBM having 256 visible nodes and 64 hidden nodes. We also consider configurations generated by a single RBM network with 1024 visible nodes and 64 hidden nodes. The factor of 4 relating the number of visible to hidden nodes is chosen to mimic the decimation of lattice sites in each step of the RG. The networks are trained on the same data used as input for the RG considered above. Training is through 10000 steps of contrastive divergence [42].

Figures 2.8a-i to 2.8a-iii show plots for the stacked RBM and Figure 2.8b for the single RBM network. Figure 2.8a-iii shows the correlation functions between the visible vectors input to the first network in the stack and the final hidden vectors output from the stack and

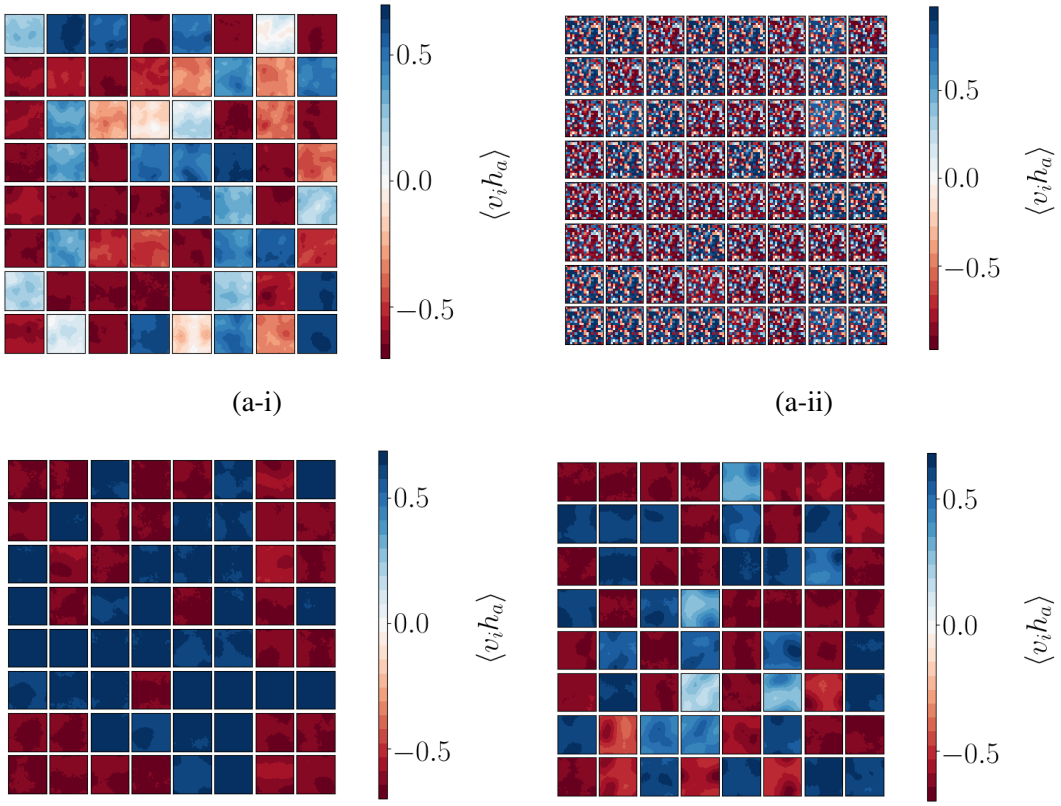


Fig. 2.8 Plots showing the correlation values for (a) the stacked RBMs various layers and (b) the single RBM. (a-i) shows correlations between visible Ising data (1024 nodes) at T_c and outputs from the first stacked RBM (256 nodes). (a-ii) shows correlations between outputs from the first stacked RBM (256 nodes) and outputs from the second stacked RBM (64 nodes). (a-iii) shows correlations between visible Ising data (1024 nodes) at T_c and outputs from the second stacked RBM (64 nodes). (b) shows correlations between visible Ising data (1024 nodes) at T_c and outputs from the single RBM (64 nodes).

is to be compared to the corresponding RG result in Figure 2.7c. The two patterns are very similar suggesting that the trained RBM is indeed performing something like the RG coarse graining.

To quantitatively compare the patterns we observe in the $\langle v_i h_a \rangle$ correlators produced by RG to those produced by the RBM we make use of a two-point correlation function. When we perform an RG coarse graining we average local nearby nodes from the input (visible lattice) to obtain the output (hidden lattice). This local averaging is encoded in the $\langle v_i h_a \rangle$ plots by bright spots. The bright spots correspond to a specific hidden node being highly correlated to a patch of local visible spins. In each $\langle v_i h_a \rangle$ plot, the hidden node we consider

is fixed and we plot its correlation with all visible nodes. If we denote each value of $\langle v_i h_a \rangle$ by x_i we calculate the two-point correlator $\langle x_i x_j \rangle$ between values $\langle v_i h_a \rangle$ and $\langle v_j h_a \rangle$ summed over all hidden nodes

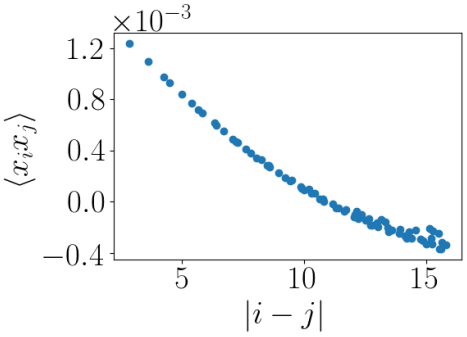
$$\langle x_i x_j \rangle = \frac{1}{N_h} \sum_{a=1}^{N_h} \langle v_i h_a \rangle \times \langle v_j h_a \rangle. \tag{2.57}$$

By calculating this quantity, we learn about the size of the correlated patches in the $\langle v_i h_a \rangle$ plots. We can plot the value of $\langle x_i x_j \rangle$ versus the distance, $|i-j|$. This quantity tells us important information about the size of the correlated patches. We average the values of $\langle x_i x_j \rangle$ where the distances $|i-j|$ are equal. The patches present in $\langle v_i h_a \rangle$ will thus be detected regardless of where they appear in the plot. If we do have local patches of high correlation, $\langle x_i x_j \rangle$ will be peaked at short distances and as distance increases, $\langle x_i x_j \rangle$ will decrease in value.

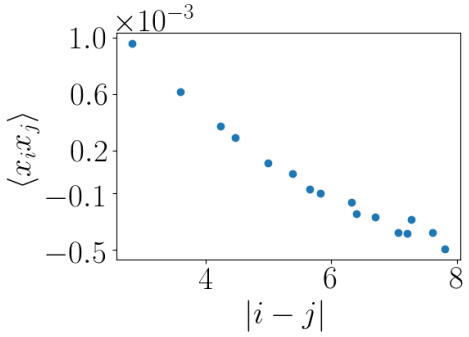
For RG, the plots seen in Figure 2.9 show a linear fall off in the correlator as distance increases. The fall off of these correlators is in the order of magnitude of 10^{-4}. In Figure 2.10 the behavior of the RBM correlator is shown. Figures 2.10a and 2.10c show similar behavior to that seen for RG. There is a linear decrease in $\langle x_i x_j \rangle$ in the same order of magnitude of 10^{-4}. A difference between these plots is that the RBM correlators are offset. We do not have an explanation for this offset.

We also study $\langle x_i x_j \rangle$ where the visible lattice is of size $48 \times 48 = 2304$ and the hidden lattice is of size $24 \times 24 = 576$. The $\langle v_i h_a \rangle$ correlators are determined using a visible set of 40000 lattices at $T_c = 2.269$ and the hidden set produced by an application of RG and by applying a trained RBM (which is trained on this same visible data set). The results for $\langle x_i x_j \rangle$ are shown in Figure 2.11. For the RBM (Figure 2.11d) the fall off of the correlator again matches the behavior of RG (Figure 2.11b). For the RBM the fall off trend is clearer with these larger lattice sizes when compared to the RBMs of smaller lattice sizes. There is a slight increase in correlation in Figure 2.11d as the distance nears $L_v/2 = 24$. This suggests that there are more than 1 local patches of correlation in the RBM $\langle v_i h_a \rangle$ plots. In Figure 2.11c we can see that some plots show some speckle with a few highly correlated spots in a single $\langle v_i h_a \rangle$ plot. We verify this observation by considering specific $\langle v_i h_a \rangle$ correlation patterns below.

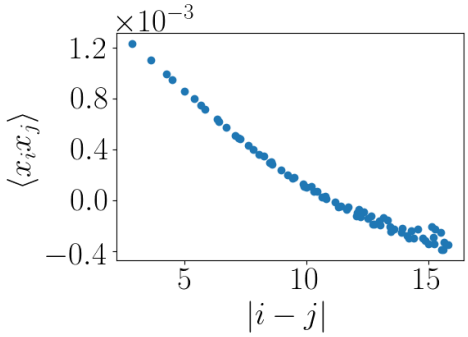
To gain more understanding of the information encoded in the two-point correlator we consider $\langle v_i h_a \rangle$ patterns of white noise in addition to a checkerboard shape with various sizes for the sub-blocks on the checkerboard. This allows us to explore the benefit of studying $\langle x_i x_j \rangle$ in probing patterns present in $\langle v_i h_a \rangle$. We show an example of a single hidden node's correlation with all visible nodes constructed using white noise in Figure 2.12a. In Figure



(a) two-point correlator versus distance for Figure 2.7a.



(b) two-point correlator versus distance for Figure 2.7b.

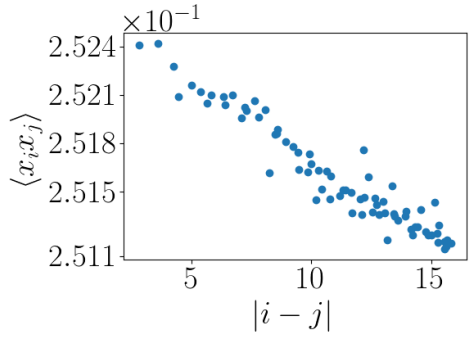


(c) two-point correlator versus distance for Figure 2.7c.

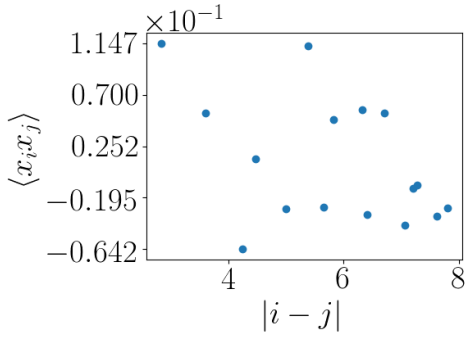
Fig. 2.9 Plots showing $\langle x_i x_j \rangle$ for RG $\langle v_i h_a \rangle$ plots shown in Figures 2.7a, 2.7b and 2.7c.

2.12b we can see $\langle x_i x_j \rangle$ calculated from the values shown in Figure 2.12a. We see different behavior to that observed in Figures 2.9 and 2.10. As expected, there is no clear relationship between the value of the two-point correlator $\langle x_i x_j \rangle$ and the distance between values x_i and x_j.

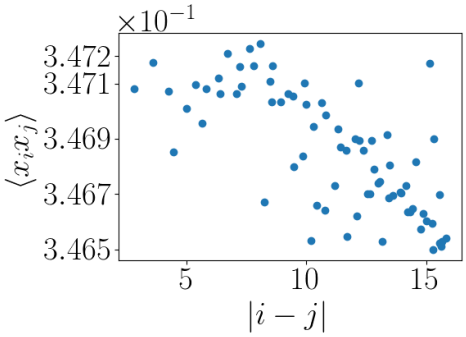
We also study $\langle v_i h_a \rangle$ with a checkerboard pattern as shown in Figure 2.13. We explore various sub-block sizes within the checkerboard pattern. In Figures 2.13a, 2.13c and 2.13e we show the $\langle v_i h_a \rangle$ plot with a checkerboard pattern on a lattice of size 32×32 with sub-blocks of size 4 by 4, 8 by 8 and 16 by 16 respectively. The corresponding two-point correlators are shown in Figures 2.13b, 2.13d and 2.13f. We can see from these plots that having many correlated patches in $\langle v_i h_a \rangle$ which are of size $< L_v/2$, produces a two-point correlator which is peaked at a number of points. In the case of Figure 2.13f, where the sub-block sizes equal $L_v/2$ we see similar behavior to that seen in the RBM and RG correlator plots. The additional peaks seen in Figures 2.13b and 2.13d are due to multiple patches in the image



(a) two-point correlator versus distance for Figure 2.8a-i.



(b) two-point correlator versus distance for Figure 2.8a-ii.

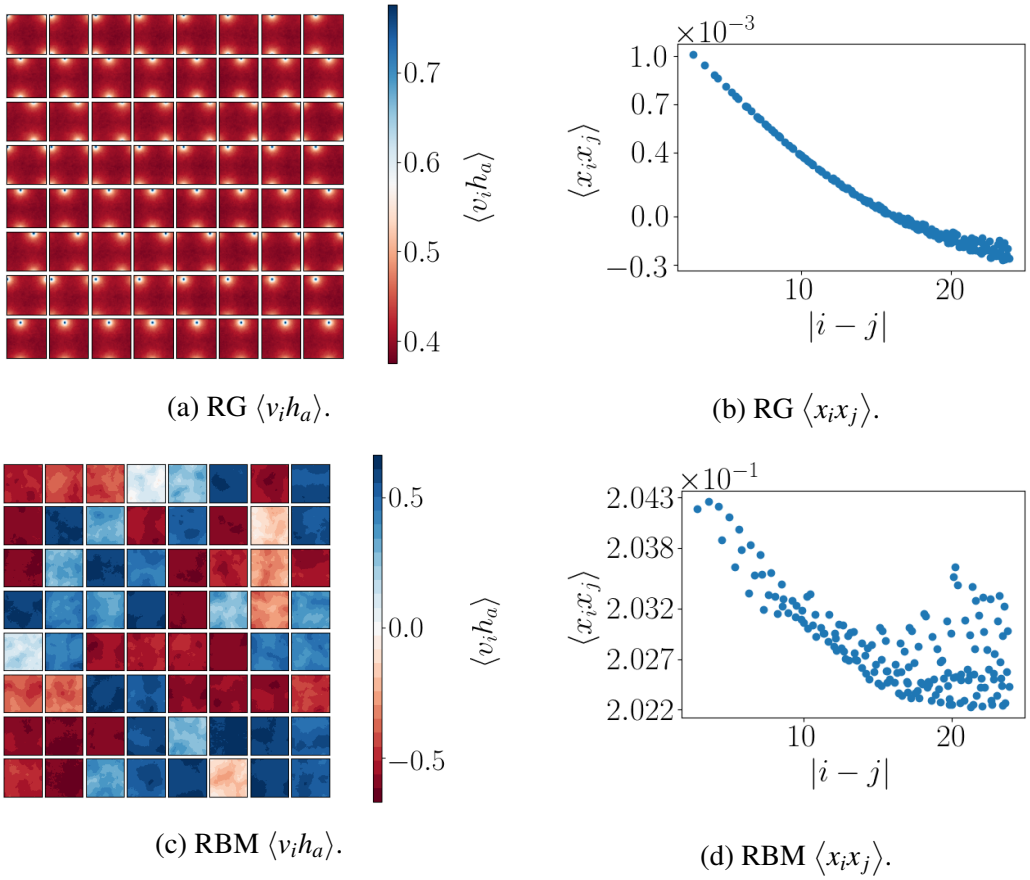


(c) two-point correlator versus distance for Figure 2.8a-iii.

Fig. 2.10 Plots showing $\langle x_i x_j \rangle$ for RBM $\langle v_i h_a \rangle$ plots shown in Figures 2.8a-i, 2.8a-ii and 2.8a-iii.

being correlated. This behavior is not characteristic of the RG local patches as a single highly correlated patch is present in the RG $\langle v_i h_a \rangle$ plots.

There is one more interesting comparison that can be carried out and it quantitatively tests the flow. The temperature is a relevant coupling so it grows as the flow proceeds. In the block spin RG that we are considering, the length of the lattice keeps halving. Thus, after 7 steps our unit of length is $2^7 = 128 \approx 100$ times larger than it was. To get some insight into the effect of this change of units, imagine we change units from centimeters to meters. In the new units, a length of 100cm is now 1m. Anything with the units of length will roughly halve with each step of the flow. In contrast to this, the temperature of the system, which in suitable units has a dimension of inverse length, will roughly double. There will be small departures from precise doubling due to interactions, but the temperature must increase by roughly a factor of 2 as each new layer is stacked. If the RBM is performing an



(a) RG $\langle v_i h_a \rangle$.

(b) RG $\langle x_i x_j \rangle$.

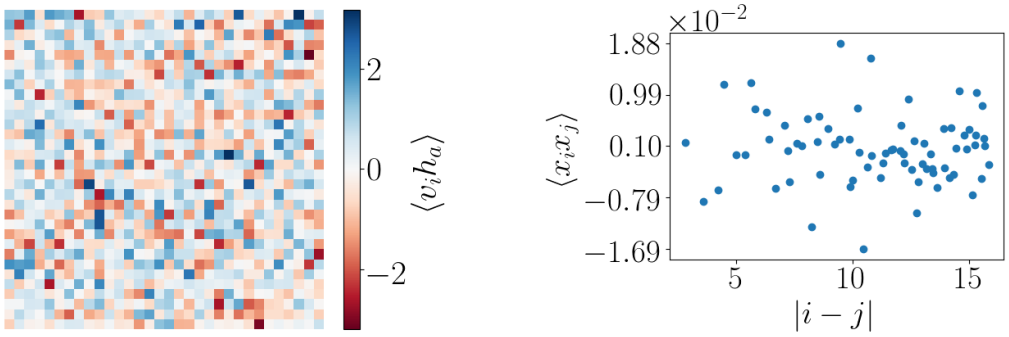
(c) RBM $\langle v_i h_a \rangle$.

(d) RBM $\langle x_i x_j \rangle$.

Fig. 2.11 Plots showing the $\langle v_i h_a \rangle$ correlators and corresponding two-point correlator $\langle x_i x_j \rangle$ values for one step of RG and a single RBM starting from an input lattice of size $48 \times 48 = 2304$ which is reduced to an output lattice of size $24 \times 24 = 576$.

RG-like coarse graining, the temperature should grow in a similar way as we pass through the layers of the deep network. Figure 2.14a plots the temperature of coarse grained lattices, generated by applying three steps of RG to an input lattice of size 64 by 64, at a temperature of $T = 2.7$. There is a clear increase in the measured temperature as the number of RG steps increase. The temperature of each layer is roughly $T = 2.3$, 4.8 and 11 for layers 1, 2 and 3 respectively, which is indeed consistent with the rough rule that the temperature doubles with each step.

Now consider a deep network made by stacking three RBMs. The first network has 4096 visible nodes and 1024 hidden nodes, the second 1024 visible nodes and 256 hidden nodes and the third 256 visible nodes and 64 hidden nodes. The network is trained on Ising data at the critical temperature, as described above. Figures 2.14b-i, 2.14b-ii and 2.14b-iii give the temperatures of the outputs of the layers of the RBM, given input lattices at temperatures of $T = 2.269$, 2 and 2.7 respectively. Temperatures of $T = 2$ and $T = 2.269$ lead to the



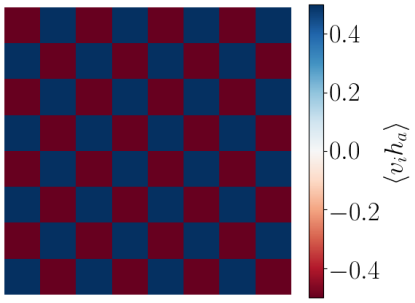
(a) (b)

Fig. 2.12 White noise: Plots showing (a) a hypothetical $\langle v_i h_a \rangle$ correlator (for a single hidden node with all visible nodes) consisting of white noise and (b) the two-point correlator $\langle x_i x_j \rangle$ calculated from the values of $\langle v_i h_a \rangle$ in (a).

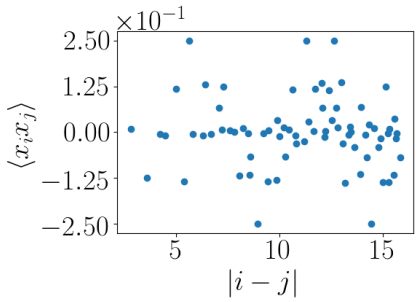
same behavior for the temperature flow, as exhibited in Figures 2.14b-i and 2.14b-ii. The temperature jumps rapidly to a high temperature in the first step of the flow, and remains fixed when the second step is taken. This is an important difference that deserves to be understood better. It questions the identification of layers of a deep network with steps in an RG flow.

Figure 2.14b-iii shows different characteristics to those of 2.14b-i and 2.14b-ii. Here the temperature of the input is above T_c at 2.7. Layer 1 is not as sharply peaked near T_c as observed in Figures 2.14b-i and 2.14b-ii. In addition to this, layers 2 and 3 are not at the same temperature but rather layer 2 is at a higher temperature than layer 3. This differs to the RG flow, where temperature increases along the flow. Figure 2.14b-iii shows a decrease in temperature from layer 2 to layer 3 rather than an increase. These plots demonstrate that the flow defined by multiple layers in a "deep" network show important differences to the RG flow. The discrepancies we have uncovered are important and precise quantitative mismatches that may provide useful clues in understanding the relationship between unsupervised deep learning by an RBM and the RG flows.

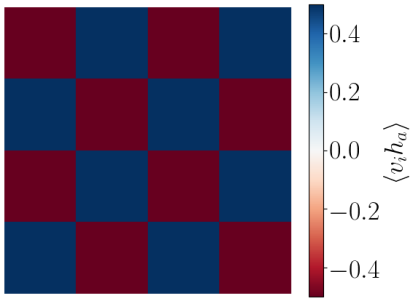
The results above have shown that the correlator $\langle v_i h_a \rangle$ exhibits RG-like characteristics. This is evident from the comparison between the $\langle v_i h_a \rangle$ plots from RG, a stacked RBM network and a network with a single RBM. We can see RG-like patterns in the correlators produced by the two RBM networks. This is a promising result that demonstrates that a form of coarse graining is taking place when networks are stacked.



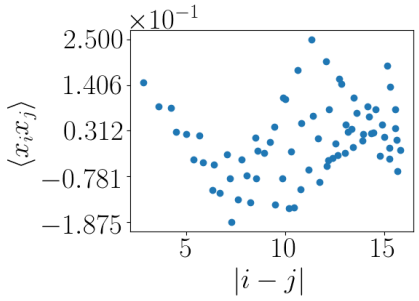
(a) $\langle v_i h_a \rangle$ grid of 32×32 nodes with sub-blocks of size 4.



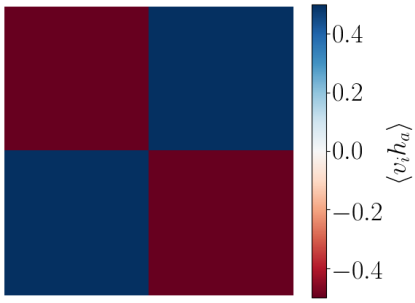
(b) $\langle x_i x_j \rangle$: sub-blocks of size 4.



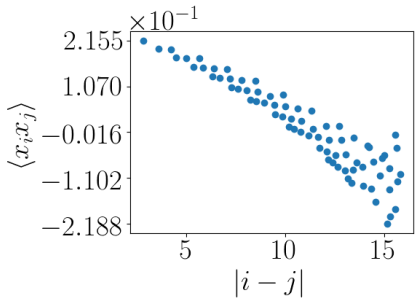
(c) $\langle v_i h_a \rangle$ grid of 32×32 nodes with sub-blocks of size 8.



(d) $\langle x_i x_j \rangle$: sub-blocks of size 8.



(e) $\langle v_i h_a \rangle$ grid of 32×32 nodes with sub-blocks of size 16.



(f) $\langle x_i x_j \rangle$: sub-blocks of size 16.

Fig. 2.13 Checkerboard: Plots showing $\langle v_i h_a \rangle$ correlators generated to depict a checkerboard with varying block sizes as well as the two-point correlator $\langle x_i x_j \rangle$ corresponding to the given $\langle v_i h_a \rangle$ plots. Plot (b) corresponds to plot (a), plot (d) corresponds to plot (c) and plot (f) corresponds to plot (e).

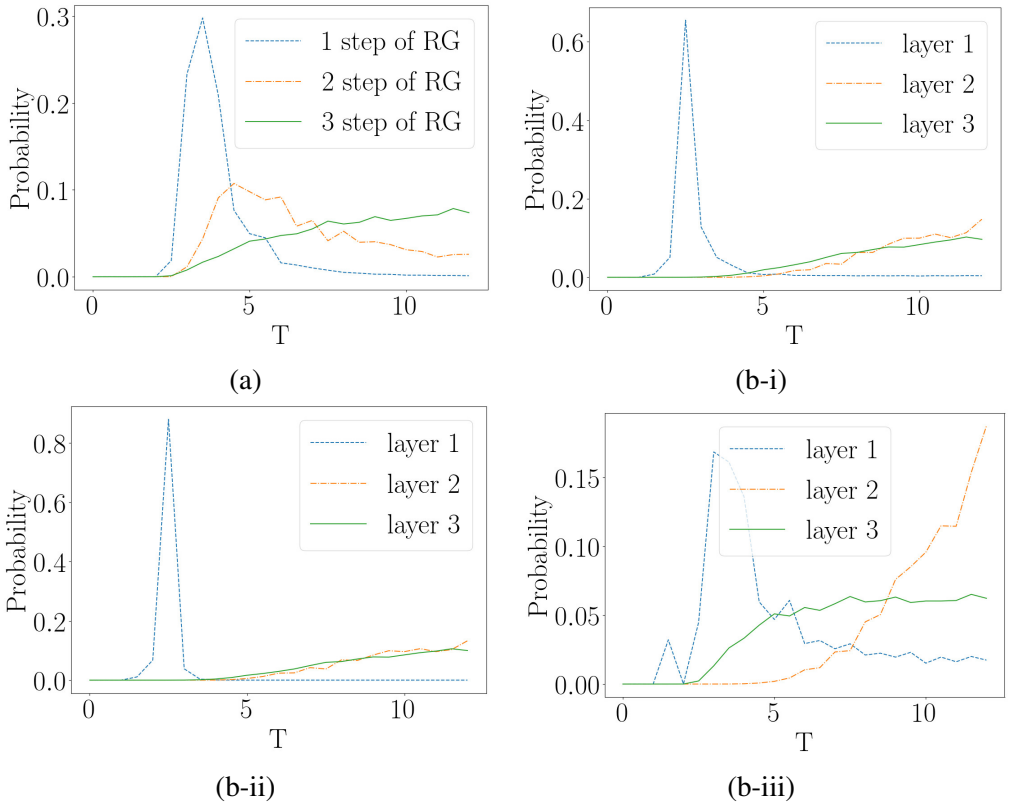


Fig. 2.14 (a) shows the average probability of the measured temperature of lattices resulting after 3 steps of RG, applied to an input lattice at T_c with 4096 sites. (b) shows the average probability plot of the measured temperature of outputs produced by a stacked RBM with 4096 input nodes, 1024 nodes in the first layer, 256 nodes in the second layer and 64 nodes in the output layer. (b-i) is given input Ising samples at $T = 2.269$, (b-ii) is given input Ising samples at $T = 2$ and (b-iii) is given input Ising samples at $T = 2.7$.

2.5 Conclusions and discussion

Our main goal has been to explore the possibility that RG provides a framework within which a theoretical understanding of unsupervised deep learning can be pursued. We have focused on a single model, the Ising model, which is naturally related to RBMs. Thus, at best our conclusions and discussion can only suggest interesting avenues for further study. We are not able to draw general definite conclusions about the applicability of RG as a framework within which a theoretical understanding of deep learning can be achieved. Our data set contains the possible states of an Ising magnet, generated using Monte Carlo simulation. This is an interesting data set, since we know that there is a well defined theory for the magnet defined on large length scales. The existence of this long distance theory guarantees that there is some emergent order for the unsupervised learning to identify. Another point worth

stressing is that the RG treatment of this system is well understood and is easily implemented numerically. It is therefore an ideal setting in which both unsupervised deep learning and RG can be implemented and their results can be compared. At the critical temperature, where the system is on the verge of spontaneous magnetization, there is an interesting scale invariant theory which is well understood. By working at this critical point, we have managed to probe the patterns generated by the RBM at different length scales and to compare it to the expected results from an RG treatment.

Our first set of numerical results compare the RBM flow introduced in [10] and further pursued in [11]. From a theoretical point of view the RBM flow looks rather different to RG since the RBM flow appears to drive configurations towards the critical temperature. The RG would drive configurations to ever higher temperatures due to the fact that the temperature corresponds to a relevant perturbation. Another important difference between the RBM flow and RG is that the number of spins is a constant of the RBM flow, but decreases with the RG flow. Our numerical results confirm that the RBM flow does indeed generate RG-like Ising configurations and we have reproduced the scaling dimension of the spin variable from the spatial statistics of the patterns generated by the RBM. This is a remarkable result and it extends and supports results reported and discussed in [10, 11]. The spin variable has the smallest possible scaling dimensions and consequently probes the largest possible scales in the pattern. When considering correlation functions of the next primary operators we find that the RBM data does not reproduce the correct scaling dimension, proving that the spatial statistics of the patterns generated by the RBM flow and those generated by RG start to differ as smaller scales are tested. We therefore conclude that the RBM flow and RG are distinct, but they do agree on the largest scale structure of the generated patterns. This is a hint into the mechanism behind the RBM flow and it deserves an explanation.

Our second numerical study has explored the idea that unsupervised deep learning is an RG flow with each stacked layer performing a step of RG. We have explained why correlation functions between the visible and hidden neurons, $\langle v_i h_a \rangle$ are capable of diagnosing RG-like coarse graining and we have computed these correlation functions using the patterns generated by the RBM. The basic signal of RG coarse graining is a "bright spot" in the $\langle v_i h_a \rangle$ correlation function, since this indicates that spins in a localized region were averaged to produce the coarse grained spin. The numerical results do indeed show a dark background with emerging bright spots. It would be interesting if the emergent patterns again guarantee agreement on the largest length scales, similar to what was found for the RBM flows, but we can not confidently make this assertion yet.

Our final numerical study considered the flow of the temperature, a relevant operator according to the RG. We find three distinct behaviors. Section 2.3.2 reviewed that RBM

flows converge to the critical temperature. This is borne out in our results. The RG flows to ever higher temperatures, with (roughly) a doubling in temperature for each step. Again, this is precisely what we observe. Finally, for a deep network made by stacking three RBMs, the temperature appears to flow when moving between the first and second layers of a deep network, but is fixed when moving between the second and third layers. This is an important difference that deserves to be understood better. It questions the identification of layers of a deep network with steps in an RG flow.

Our results are encouraging. There are enough similarities between unsupervised learning by an RBM and the RG flow that the relationship between the two should be developed further. Regarding future studies, it maybe useful to explore models other than Ising. The Ising model has an unstable fixed point due to the presence of relevant operators. Consequently, finite flows starting near the critical point all terminate on different models. In this case its not easy to know if the RBM has flowed to the "right answer" because there are many possible right answers! The stable fixed point of the model is at infinite temperature and the configurations at this fixed point are random with correlators that have a correlation length of zero. This is hardly a promising answer to shoot for. It maybe more instructive to study models that have an attractive RG fixed point. In this case the minimum that the RBM is looking for would be unique and the connection between the two may be easier to recognize.

We have in mind systems that exhibit self organized criticality [50], including models constructed to understand the spread of forest fires [48] and models for the spread of infectious diseases [49].

By using Ising model data, generated by Monte Carlo simulation, starting from a local Hamiltonian we know how a coarse graining capable of identifying emergent patterns should proceed: spatially neighboring spins should be averaged. For more general data sets, this may not be the case. It is fascinating to ask what the rules determining the correct coarse graining are and in fact, with respect to this question, unsupervised deep learning has the potential to shed light on RG.

Another interesting comparison worth mentioning is the similarity between an average pooling layer within a convolutional neural network (CNN) and the averaging performed in variational RG. CNNs are known for their excellent performance in image recognition and classification tasks [4, 55, 56]. CNNs have a number of layers which act on groups of nearby pixels in the image. One of these layers which is similar to the coarse graining performed in variational RG is called a pooling layer. The pooling layer performs a down-sampling on the data it receives from previous layers in the network [57–59]. The down-sampled data is more robust to changes in position of features and gives the network the property of local translation invariance. One way in which the pooling operation is implemented is by

averaging all values in the given patch of data it acts on to obtain a new value to replace these values. Pooling usually averages blocks of data which are of size 2×2. This results in an input block of data being reduced by a factor of 2 in length and by a factor of 4 in the number of values which is the same factor of rescaling which occurs in variational RG.

In recent years a connection between the renormalization group and tensor networks [60] has been discovered, providing a connection to the field of quantum information. The discovered connection demonstrates that the multi-scale entanglement renormalization ansatz (MERA) tensor networks carry out a coarse graining that agrees in many ways with the coarse graining performed by the renormalization group [61] . This suggests that there maybe a link between tensor networks and deep learning. For related ideas see [62]. Since tensor networks have been extensively studied for calculations the connection may prove to be useful for better understanding deep learning.

Apart from the exciting possibility that the link to RG might contribute towards a theoretical understanding of unsupervised deep learning, one might also ask if the connection would have any practical applications. One possibility that we are currently pursuing, is a Callan-Symanzik like equation governing the learning process. Roughly speaking, one might mimic RG by dividing the weights to be learned into relevant, marginal and irrelevant parameters, depending on gross statistical properties of the training data. If this classification is itself not too expensive, one could pursue a more efficient approach towards training, since the classification of weights would provide an understanding of which weights are important, and which can simply be set to zero. We hope to report on this possibility in the future.

2.6 Author's contribution

The above work is published in IEEE Access and is a collaboration between the author, Robert de Mello Koch and Ling Cheng.

The author had a major contribution to the above published article which includes the development and implementation of all numerical experiments and contributions to writing the manuscript.

Chapter 3

Short sighted deep learning

3.1 Introduction

In recent years machine learning has shown remarkable success in a wide range of problems. These algorithms can outperform humans at a number of tasks including image classification and complex strategy games [4, 63, 44, 43, 22, 23]. In many fields efforts to harness the power machine learning offers are being pursued. Specifically in physics, machine learning is being explored as a tool to uncover properties of systems that are difficult to study analytically [34, 64–68, 35, 69]. However, there is at present no convincing explanation of how these networks learn and why they can achieve such remarkable feats [25, 27, 18, 29].

Recent studies have aimed to gain insight into how and why deep learning works, by employing the statistical mechanics of spin systems as simple toy models [9, 10, 70–72, 11, 32, 31]. The idea is that deep learning, which is a form of coarse graining, is related to the renormalization group (RG) of statistical mechanics and quantum field theory. This is a natural idea since both problems are concerned with reducing many degrees of freedom to an effective description employing far fewer degrees of freedom. Spin systems are an ideal setting in which to explore this proposal, thanks to their simplicity. In addition to the simplicity of these models, their statistical mechanics is well understood making them a reasonable laboratory for deep learning studies. Further, spin systems are also not far removed from more standard applications of deep learning including image analysis: the state of a spin system is composed of local interacting patches much in the same way that an image is made up of local collections of pixels which are highly correlated [36].

Although a number of recent studies have explored the link between Kadanoff's variational renormalization group (RG) and unsupervised deep learning [9, 10, 70–72, 11, 32, 31], it is still not settled whether such a link exists. Studies have so far focused on nearest neighbor

Ising models. We add to this discussion by considering long range interacting spin systems which, as we will see, exhibit important differences from their short ranged cousins.

The paper [10] trained three (Restricted Boltzmann machines) RBM networks on Ising model data. The first was trained on data generated at temperatures ranging from above to below the critical temperature, the second network on data above the critical temperature and the third at temperatures below the critical temperature. The trained weights of each network were used to define a map from the visible spins back to the visible spins. Iterating this map defines an RBM flow. Results produced by the first network show interesting behavior suggesting that RBMs learn the critical temperature in the sense that the RBM flow terminates on the critical point of the Ising model. This is in contrast to RG flows: the critical point of the Ising model is an unstable fixed point and fine tuning is required to construct an RG flow trajectory that terminates on this critical point.

In a follow up paper, [70], scaling dimensions are used to provide precise quantitative tests that probe whether or not the network reproduces the critical point dynamics. The approach recognizes that the critical point of the Ising model is described by a conformal field theory (CFT), so that correlation functions of the spins in patterns generated by the RBM can be compared to known CFT predictions. The results of [70] show that the network correctly reproduces the smallest scaling dimension, which belongs to the Ising spin itself. The network does not correctly reproduce the next smallest scaling dimension, which is related to the energy density. This implies that although the RBM has correctly learned the largest scale correlations in the critical spin states, it fails on shorter length scales.

Our goal is to repeat the above investigations, for a two-dimensional long range spin lattice. Training a network on a system with a different range of interactions will allow us to probe how robust the claim is that RBMs learn the critical temperature of a given spin system and we can again probe what length scales are captured by training. We use Markov Chain Monte Carlo methods (MCMC) to determine the critical temperature of the long range spin lattice and corresponding scaling dimensions, as well as to generate the RBM training data. Our results suggest that the RBM flow does not converge to the critical temperature. Further, stacking RBMs does not appear to generate a flow that matches the coarse features of RG flows. Consequently, one lesson we learn is that it is dangerous to draw general conclusions from numerical experiments conducted on a single system. Furthermore an interesting question which suggests itself (but which we will not manage to answer) is the question of what classes of models a given conclusion applies to.

The long range spin lattice we study is described in Section 3.1.1. A quick overview of RG and RBMs is given in sections 3.1.2 and 3.1.3. MCMC is discussed in Section 3.1.4

along with the numerical approximations made for various properties of the long range spin lattice. The results of our study are presented in Section 3.2.

3.1.1 Long range spin lattice model

The long range spin lattice is a model of a magnet made up of binary spins. Considering the two-dimensional model we have a square lattice of sites with a spin at each site. Each spin, $s_{\mathbf{k}}$, can take values of ± 1 and each site $\mathbf{k}$ is labelled by a two-dimensional vector of integer coordinates $\mathbf{k} = (x, y)$. Spins on sites $\mathbf{i}$ and $\mathbf{j}$ interact with a strength of $J_{\mathbf{ij}}$. Each spin also interacts with an external magnetic field, with strength μ.

The Hamiltonian of the long range spin lattice model is given by

$$H = -\sum_{\langle \mathbf{i},\mathbf{j} \rangle} J_{\mathbf{ij}} s_{\mathbf{i}} s_{\mathbf{j}} - \mu \sum_{\mathbf{j}} s_{\mathbf{j}}. \tag{3.1}$$

The interaction strength $J_{\mathbf{ij}}$ is inversely proportional to a power of the distance between sites $\mathbf{i}$ and $\mathbf{j}$

$$J_{\mathbf{ij}} = \frac{1}{r^{\alpha}}, \tag{3.2}$$

where $r = |\mathbf{i} - \mathbf{j}|$. We set $\alpha = 3$ and $\mu = 0$.

For the two-dimensional nearest neighbour Ising model, analytic results give precise values for the critical temperature and scaling dimensions. When moving to models such as the long range spin lattice, no analytic results exist and we thus rely on MCMC methods to determine these properties before training the RBM [73].

The critical point of the long ranged spin system will again be described by a CFT. The two-point functions of primary operators of this CFT will have the usual power law fall off, something that can be explored numerically. The two-point function of primary operators at the fixed point of the spin lattice are of the form [51]

$$\langle \phi(\vec{x}_1)\phi(\vec{x}_2) \rangle = \frac{B}{|\vec{x}_1 - \vec{x}_2|^{2\Delta}}, \tag{3.3}$$

where Δ is the scaling dimension and B is a constant. To probe the largest scales of the patterns generated by the RBM we will focus on the two operators of the lowest dimension. The first operator is the spin itself. The two-point correlator between spins $\langle s_{\mathbf{i}} s_{\mathbf{j}} \rangle$, with scaling dimension Δ_s takes the form

$$\langle s_{\mathbf{i}} s_{\mathbf{j}} \rangle = \frac{B_s}{|\mathbf{i} - \mathbf{j}|^{2\Delta_s}}, \tag{3.4}$$

with Δ_s to be determined numerically. The second operator of interest is the energy density where we study the correlator between $\langle \varepsilon_{\mathbf{i}} \varepsilon_{\mathbf{j}} \rangle$, with scaling dimension Δ_ε

$$\langle \varepsilon_{\mathbf{i}} \varepsilon_{\mathbf{j}} \rangle = \frac{B_\varepsilon}{|\mathbf{i}-\mathbf{j}|^{2\Delta_\varepsilon}}, \tag{3.5}$$

with

$$\varepsilon_{\mathbf{i}} = s_{\mathbf{i}} \sum_{\langle \mathbf{i},\mathbf{j} \rangle} J_{\mathbf{ij}} s_{\mathbf{j}} - \bar{\varepsilon}_{\mathbf{i}}, \tag{3.6}$$

$$\bar{\varepsilon}_{\mathbf{i}} = \frac{1}{N_s} \sum_{k=1}^{N_s} \varepsilon_{\mathbf{i}}^{(k)}, \tag{3.7}$$

where N_s is the number of sample configurations summed and (k) denotes the kth sample. We use MCMC sampling methods to determine the critical temperature of this model which is found to be $T_c \approx 7.7$ (details are discussed later in Section 3.1.4).

3.1.2 RG

RG defines a coarse graining transformation which is applied repeatedly to a microscopic description of a system to eventually arrive at a macroscopic description of the system [74, 75]. Each application of RG results in a rescaling where unimportant degrees of freedom are removed and important degrees of freedom remain. The degrees of freedom or operators are characterized as either relevant, irrelevant or marginal. At each step of applying RG, couplings between variables change and we obtain a new coarse grained Hamiltonian [74]. The only change that occurs when obtaining a new Hamiltonian at each step is the couplings update. The couplings that are relevant will become larger and the couplings that are irrelevant will get smaller as more steps of coarse graining are applied. The marginal terms in the Hamiltonian will remain unchanged.

In this thesis we study a discrete spin lattice and so are interested in a discrete version of RG, defined by Kadanoff, which performs a block spin average [76]. Block spin averaging takes four spins as seen in Figure 3.1 in red, and averages these spins to produce the blue spin found in the centre of each group of four red spins. The average value of the red spins becomes the value of the new centre blue spin. By performing this averaging we rescale lengths in the system by a factor of $1/2$.

The RG flow halts at a critical point where the system exhibits scale invariant properties. This is because all couplings remain unchanged regardless of the length scale studied [51]. A basic observable of the fixed point is the spectrum of scaling dimensions.

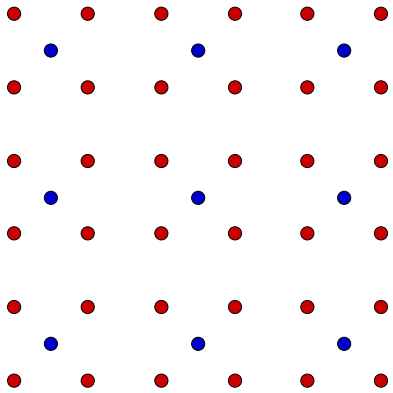


Fig. 3.1 Block-spin averaging as introduced by Kadanoff. Groups of four red sites are averaged to obtain blue sites within their centre.

3.1.3 RBM

Restricted Boltzmann machines are made up of two layers of nodes, an input layer called the visible layer and an output layer called the hidden layer. The number of nodes in the visible layer is dependant on the number of data points each training sample has. Selecting the number of nodes in the hidden layer is an iterative process, which tries to select architectures that give the best results. Nodes in both layers take values of ± 1. Every visible node is connected to every hidden node and connections between nodes in the same layer are forbidden. The connection between a visible node v_i and a hidden node h_a has an associated weight W_{ia}. Each visible node v_i and hidden node h_a has an associated bias $b_i^{(v)}$ and $b_a^{(h)}$ respectively.

The energy function of an RBM is given by

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i=1}^{N_v}\sum_{a=1}^{N_h} v_i W_{ia} h_a - \sum_{i=1}^{N_v} v_i b_i^{(v)} - \sum_{a=1}^{N_h} h_a b_a^{(h)}, \tag{3.8}$$

where N_v is the number of visible nodes and N_h is the number of hidden nodes. This energy determines the probability of a visible and hidden vector occurring together, which is given by

$$P(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}, \tag{3.9}$$

where

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}, \tag{3.10}$$

is the partition function. The goal of training is to match the distribution of the input data $q(\mathbf{v})$ to the distribution generated by the RBM model $p(\mathbf{v})$ [41], where

$$p(\mathbf{v}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}. \tag{3.11}$$

To achieve this the weights and biases are updated using derivatives of the Kullback-Liebler divergence

$$\begin{aligned} D_{KL}(q||p) &= \sum_{i=1}^{N_v} q(v_i)\left(\log\left(q(v_i)\right) - \log\left(p(v_i)\right)\right) \\ &= \sum_{i=1}^{N_v} q(v_i)\log\left(\frac{q(v_i)}{p(v_i)}\right). \end{aligned} \tag{3.12}$$

Computation of the partition function Z is practically impossible due to the enormous number of states summed over. To overcome this an approximate method, contrastive divergence, is used to train the network [39, 41, 46]. Rather than summing over the entire space of vectors, we sum only over the vectors given in the training set [42]. During training the weights and biases are updated according to the following update rules

$$\frac{\partial D_{KL}(q||p)}{\partial W_{ia}} = \langle v_i h_a\rangle_{data} - \langle v_i h_a\rangle_{model}, \tag{3.13}$$

$$\frac{\partial D_{KL}(q||p)}{\partial b_i^{(v)}} = \langle v_i\rangle_{data} - \langle v_i\rangle_{model}, \tag{3.14}$$

$$\frac{\partial D_{KL}(q||p)}{\partial b_a^{(h)}} = \langle h_a\rangle_{data} - \langle h_a\rangle_{model}. \tag{3.15}$$

We use values from the training data set to determine expectations given by $\langle\cdot\rangle_{data}$, and values from the model to determine expectations given by $\langle\cdot\rangle_{model}$. To obtain the expectation of $\langle v_i\rangle_{data}$ we simply use the training data set, however we do not have a set of hidden vectors in the training set to determine $\langle v_i h_a\rangle_{data}$ or $\langle h_a\rangle_{data}$. In order to find a set of data hidden vectors which we can use in the data expectations we sample hidden vectors using the training data set with a probability given by equation (B.10)

$$p(h_a|\mathbf{v}) = \frac{1}{2}(1+\tanh(\sum_i v_i W_{ia} + b_a^{(h)})). \tag{3.16}$$

We also do not have a set of visible vectors and hidden vectors for the model expectation values. We use the data hidden vectors (generated using equation (B.10) and the training

data set) to generate model visible vectors with equation (B.11)

$$p(v_i|\mathbf{h}) = \frac{1}{2}(1+\tanh(\sum_a h_a W_{ia} + b_i^{(v)})). \tag{3.17}$$

Then using these model visible vectors, we can generate a set of model hidden vectors using equation (B.10). This occurs at each step of training in order to calculate the updates given by equations (3.13), (3.14) and (3.15).

Once training is complete, we can generate configurations, which we call an RBM flow, using the trained RBM [10, 11]. Starting from a set of visible vectors $\mathbf{v}^{(0)}$ (could be the initial training data or MCMC data at a specific temperature), we generate a flow of visible and hidden vectors by sampling with the probabilities given in equations (B.10) and (B.11), each time using the most recently generated set of vectors

$$\mathbf{v}^{(0)} \to \mathbf{h}^{(0)} \to \mathbf{v}^{(1)} \to \mathbf{h}^{(1)} \to \mathbf{v}^{(2)} \to \cdots . \tag{3.18}$$

In order to get $\mathbf{h}^{(0)}$ we would sample hidden nodes given the visible vectors $\mathbf{v}^{(0)}$. To get $\mathbf{v}^{(1)}$ we would sample visible nodes given the hidden vectors $\mathbf{h}^{(0)}$. We can iterate the process to generate an RBM flow of a certain length. Flow length is defined as the number of visible samples generated using the visible MCMC data set $\mathbf{v}^{(0)}$ and hidden set $\mathbf{h}^{(0)}$. The hidden set $\mathbf{h}^{(0)}$ is found using equation (B.10) given the trained RBM weights and biases and the visible training data. $\mathbf{v}^{(1)}$ can then be found using equation (B.11) given $\mathbf{h}^{(0)}$ and the trained RBM weights and biases. In equation (3.18), $\mathbf{v}^{(1)}$ would define a flow of length 1 and $\mathbf{v}^{(2)}$ would define a flow length of 2. We could continue this chain and with each successive $\mathbf{v}^{(n)}$ generated, the flow length increases by 1. In Section 3.2.1 we discuss results derived from RBM flows.

3.1.4 Markov chain Monte Carlo

Studying the long range spin lattice analytically is difficult. We rely on MCMC numerical simulations to determine key statistical properties of the fixed point of the long range spin lattice. The fixed point theory is a conformal field theory, with a special class of operators, primary operators, that have two point functions that have a power law fall off. Probing these correlation functions provides detailed quantitative tests for the RBM output. We are most interested in two primary operators: the basic spin and energy density operators given in equations (3.4) and (3.5).

We begin with a randomly initialized lattice with each spin taking values of ± 1 and a fixed temperature. A single step of MCMC consists of the following:

1. Randomly select a site $\mathbf{i}$.
2. Determine the change in energy, ΔE, that results from flipping $s_{\mathbf{i}}$.
3. If $\Delta E \leq 0$, accept the new configuration with probability 1.
4. If $\Delta E > 0$, accept the new configuration with probability $e^{-\Delta E/T}$.

We can generate a chain of lattice states by repeating the above process. Starting from a randomly initialized lattice means that a number of MCMC steps are required before we have a state that resembles states at the temperature we are concerned with. To ensure we sample valid states, allow a burn in period of sufficiently many MCMC steps before keeping samples generated. The number of steps required for the burn in period is not known precisely but a good rule of thumb is to measure observables such as the magnetization or specific heat of the samples generated and ensure these values have stabilized before keeping samples generated.

We generate 2000 samples at each temperature $T = 0, 0.1, 0.2, \ldots, 14$ using a burn in period of 50000 MCMC steps. From the magnetization plot in Figure 3.2c we see the critical temperature lies between $T = 5$ and $T = 8$.

We can give an independent determination of the critical temperature by studying correlation functions of the primary operators given in equations (3.4) and (3.5). The scaling dimension related to the energy, Δ_ε, must have a value of 1 at the critical point. From Figure 3.2a we see that $\Delta_\varepsilon = 1$ when $T = 7.7$. This value for the critical temperature lies in the correct range given by the magnetization plot. Using 7.7 for T_c we determine Δ_s from Figure 3.2b which is shown by the vertical and horizontal lines as 0.53.

3.2 Numerical results

We perform two separate numerical experiments - one investigating a single RBM network and one investigating a simple "deep" network. The first experiment studies the RBM flow, while the second explores the possibility that the layers in a stacked RBM network mimic steps in an RG flow.

The RBM flow is generated using a trained RBM. The flow produces sets of visible vectors, one for each given initial vector, as described in Section 3.1.3. Our RBM flows for the two-dimensional long range spin lattice can be compared to existing results for the two-dimensional Ising spin lattice with nearest neighbour interactions. As we will see, this comparison sheds light on the possible connection between RBM and RG flows. Existing work shows that for the two-dimensional nearest neighbour Ising model, the RBM flows

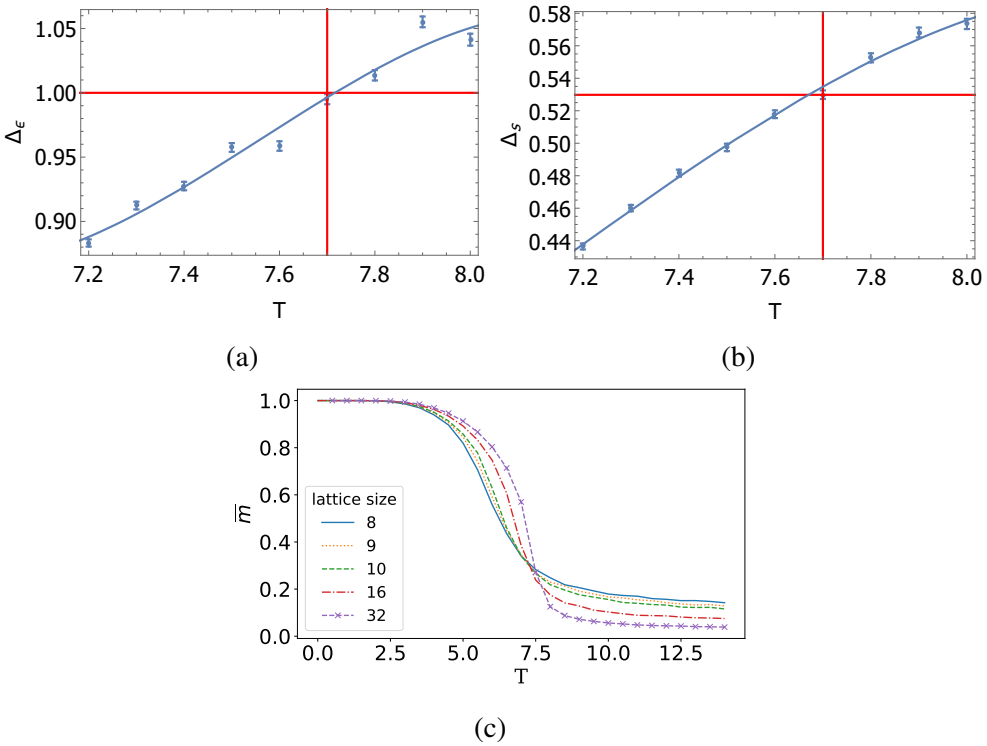


Fig. 3.2 Scaling dimensions of the spin lattice, with Hamiltonian as in equation (3.1) with $\mu = 0$ and $\alpha = 3$, determined using MCMC. Δ_ε is 1 and $\Delta_s = 0.53$ at $T = 7.7$. These values are at the intercept of the vertical and horizontal red lines in plots (a) and (b). (a) shows Δ_ε versus temperature for s lattice of size of 10 by 10. (b) shows Δ_s versus temperature for a lattice size of 10 by 10. (c) shows the magnetization versus temperature curve for various lattice sizes.

converge to the critical temperature of the system [10, 11]. This is in contrast to RG flows because the Ising critical point is unstable and significant fine tuning is needed to ensure that a flow will terminate at the critical point. The attractive fixed points for the 2D Ising model lie at high and low temperature. Any points near an attractive fixed point will flow towards this point. A more interesting fixed point lies at the critical temperature T_c and is unstable. This fixed point is not attractive but rather called a mixed point [37]. For some points near a mixed point one will flow towards the fixed point and for others one will flow away from it. Here the temperature must be tuned very carefully to ensure that no relevant operators are captured and we flow towards the critical point. If we do capture even a small piece of a relevant operator, the application of RG will result in this operator growing and thus we flow away from the fixed point. Consequently we almost always flow away from the critical point and this is why it is regarded as an unstable fixed point. The fixed point lying at the

critical temperature defines where the Ising model experiences a phase transition. Here the two competing phenomena of minimizing energy and maximizing entropy exactly balance.

Do RBM flows of the long range spin lattice flow to the critical point? We have interrogated this question by comparing scaling dimensions for the spin and energy density operators, as well as temperatures, calculated using the configurations generated by the RBM flow. By calculating these quantities for successive configurations in an RBM flow we determine if there is a fixed point of the flow and whether or not this fixed point is related to critical point of the long range spin lattice. The results obtained from the RBM flows are reported in Section 3.2.1.

The second experiment chooses the number of nodes in the stacked RBM networks in such a way that if there is a correspondence to RG, the block spinning of each step combines spins in groups of 4. This corresponds to scaling the input lattices by a factor of $1/2$. To accomplish this, choose the number of hidden nodes to be $1/4$ of the number of visible nodes. We study the flow generated by each RBM layer. Each layer of the RBM network produces a set of hidden vectors that are input to the next RBM network in the stack. Comparing each RBMs hidden layer configurations to steps in the coarse grained configurations produced by RG gives a quantitative comparison between the two methods.

The stacked network of RBMs is trained in a greedy layer-wise manner [28]. In greedy layer-wise training, each layer is trained independently and the hidden vectors produced by a trained layer are used as training data for the next layer. Once all layers have been trained, we use the initial set of visible configurations (generated from MCMC at T_c) and the hidden configurations from each layer to calculate correlations between visible and hidden nodes. This $\langle vh \rangle$ correlator provides a diagnostic for comparing the coarse graining performed by RG versus that performed by stacked RBM networks.

For the variational RG $\langle vh \rangle$ correlators for v configurations we use the initial set of configurations generated by MCMC, while the h configurations are generated by block spinning. Results comparing RG to the stacked RBM are reported in Section 3.2.2.

3.2.1 RBM flows

In this experiment we train a single RBM network and study the associated RBM flow. An RBM flow is a sequence of configurations generated by the trained network, starting from a set of initial visible configurations (see Section 3.1.3 for more details). The RBM network we consider has $N_v = 10 \times 10 = 100$ visible neurons and $N_h = 9 \times 9 = 81$ hidden neurons.

The data used to train the network is generated using MCMC using the long range spin lattice Hamiltonian. 2000 configurations are generated at each temperature $T = 0, 0.5, \ldots, 14$. The network is trained using 30000 training steps, a learning rate of 10^{-3} and batches of

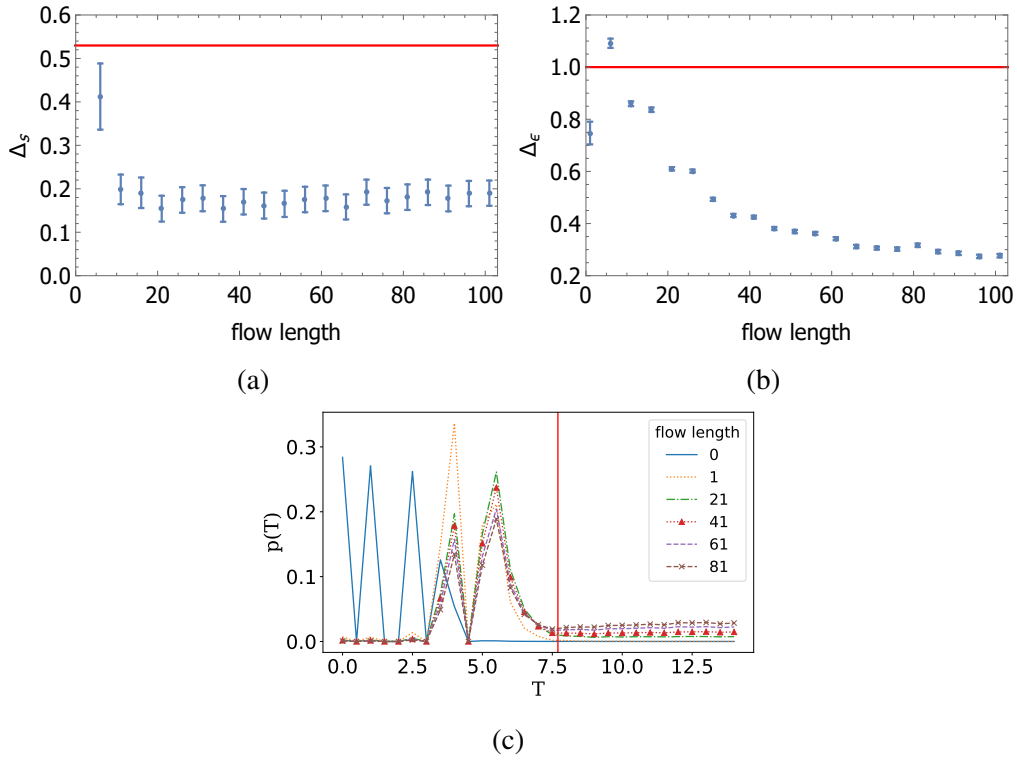


Fig. 3.3 RBM flows starting from low temperature ($T = 0$) configurations. The RBM has $N_v = 100$ visible nodes and $N_h = 81$ hidden nodes. The training set is comprised of 2000 configurations at each temperatures $T = 0, 0.5, \ldots, 14$ giving 30000 configurations in total. (a) shows Δ_s versus flow length. (b) shows Δ_ε versus flow length. (c) shows the average probability of temperature versus temperature of flows of various lengths. The flow tends to a temperature of 6. The red vertical line indicates the critical temperature of 7.7.

1000 samples. Once the network has been trained, we generate flows starting from (i) low temperature $T = 0$, (ii) the critical temperature $T_c = 7.7$ and (iii) high temperature $T = 14$. The configurations generated for different flow lengths determine the scaling dimensions of the spin, Δ_s, and energy density, Δ_ε, operators. In addition, by employing a supervised neural network, we also measure the average temperature of flows at different lengths. This allows us to determine both how the dimensions and the temperature evolve with the RBM flow. The supervised network is trained on the same data as is used to train the RBM. Note however that for the supervised case this data contains temperature labels.

Results for flows starting from (i) low temperature configurations are given in Figure 3.3, (ii) the critical temperature in Figure 3.4 and (iii) high temperature in Figure 3.5. Error bars shown on these plots are determined using Mathematica's NonlinearModelFit function. Using the standard error obtained from the regression, the Student's - t distribution is used to calculate a confidence interval for the given parameters with a 95% confidence level. These

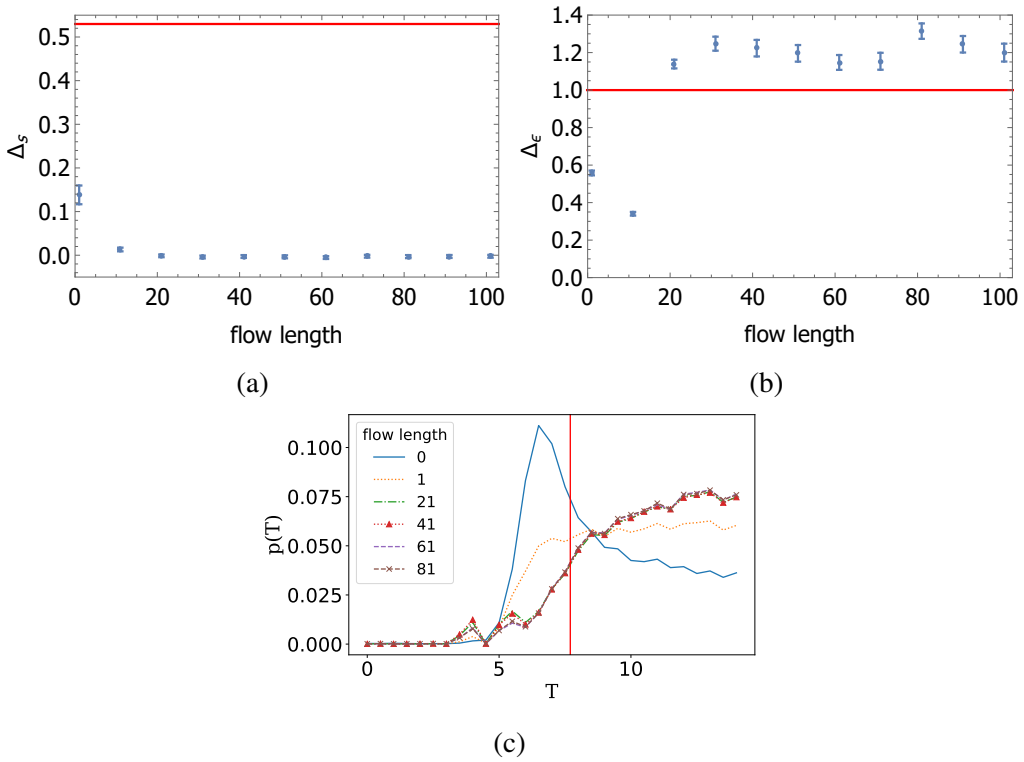


Fig. 3.4 RBM flows starting from configurations near the critical temperature ($T_c \approx T = 7.7$). The RBM has $N_v = 100$ visible nodes and $N_h = 81$ hidden nodes. Training uses 2000 configurations at temperatures $T = 0, 0.5, \ldots, 14$ giving 30000 configurations in total. (a) shows Δ_s versus flow length. (b) shows Δ_ε versus flow length. (c) shows the average probability of temperature versus the temperature of flows of various lengths. The red vertical line indicates the critical temperature of 7.7.

results are found using 30000 samples, which are generated by performing the RBM flow, using a single trained RBM network. For each flow length shown in the plot, we use the value of $\langle v_i v_j \rangle$ calculated from 30000 samples and the corresponding distances $|i-j|$. This allows us to determine a value for Δ_s and Δ_ε with a 95% confidence interval for a given flow length.

Figure 3.3a converges after a flow length of ± 20 to a value for Δ_s of approximately 0.175 which underestimates the value determined from MCMC of 0.53 shown by the red horizontal line. This implies that the patterns generated by the RBM are more correlated on large length scales than they should be. In contrast to this, results obtained in [70] show that flows generated by an RBM trained on two-dimensional Ising model configurations, starting from low temperature configurations, converge to the correct value for Δ_s [70]. There is not improvement when these RBM configurations are used to estimate the scaling dimension of the energy density: Figure 3.3b shows that the flows generate a value of $\Delta_\varepsilon = 0.25$, which is

again an under estimate of the correct value $\Delta_\epsilon = 1$. The RBM consistently underestimates the spin scaling dimension, implying that distant spins are more correlated than they should be. The RBM flows have a scaling dimension which is smaller than the value determined by MCMC simulations. Having a smaller scaling dimension implies that the two-point correlation values will have a slower fall off as distance increases. The network has learnt that spins far away from each other are more correlated than they are in reality. Intuitively, more correlation suggests that we are below the critical temperature and we indeed find that the flows converge to a temperature of approximately 6 which is below T_c. A configuration below the critical temperature, has more correlation between distant spins than those at the critical temperature. In [70] for the case of the Ising model, the spin scaling dimension is correctly determined from the RBM flows. In this case no long range interactions exist.

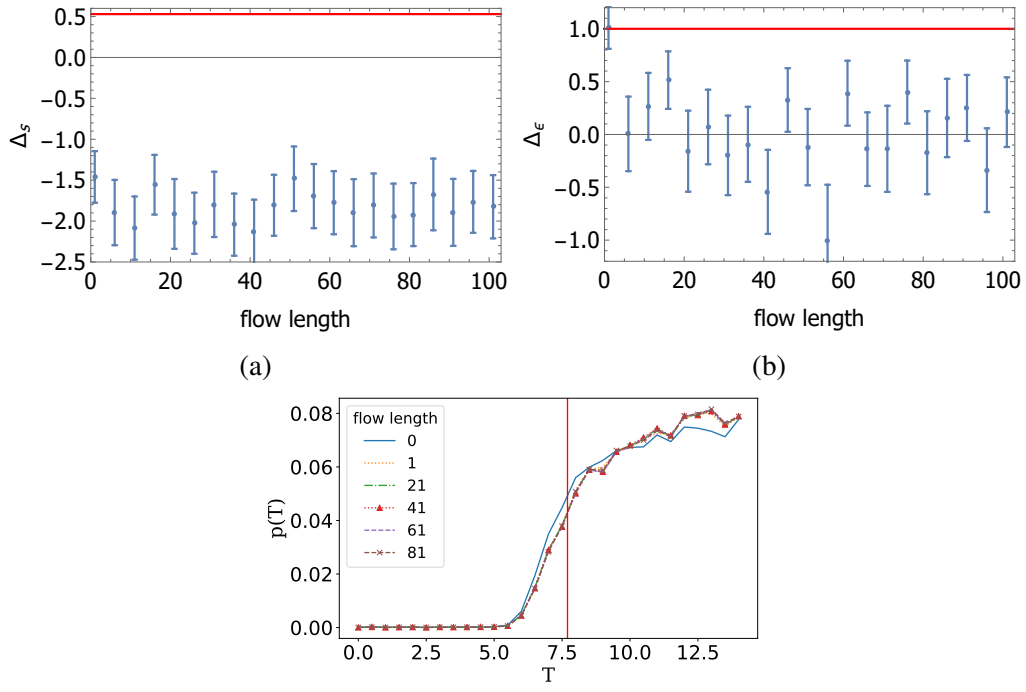


Fig. 3.5 RBM flows starting from configurations at a high temperature ($T = 14$). The RBM network has $N_v = 10 \times 10 = 100$ visible nodes and $N_h = 9 \times 9 = 81$ hidden nodes. The training set is made up of configurations at temperatures of $T = 0, 0.5, \ldots, 14$ with 2000 configurations at each temperature (in total 30000 configurations). (a) shows Δ_s versus flow length. (b) shows Δ_ϵ versus flow length. (c) shows the average probability of temperature versus temperature of flows of various lengths. The red vertical line indicates the critical temperature of 7.7.

Clearly the RBM has not correctly determined the critical point of the long range spin lattice, which is in tension with conclusions reached when studying the nearest neighbor Ising spin lattice: in that case flows converge to the critical temperature [10, 11].

Starting the flow from configurations at the critical temperature gives the results shown in Figure 3.4a. The flows converge to a value close to 0 for Δ_s, well below the critical point value. For Δ_ε the flow converges to a value near 1.2, as shown in Figure 3.4b. This is an over estimate of the critical point value $\Delta_\varepsilon = 1$. Figure 3.4c shows that the flow converges to a temperature of 3.75, again lower than the critical temperature. These results suggest that the configurations generated by the RBM flow have not managed to capture the physics of the long ranged spin lattice.

For the RBM flow starting from high temperature, results in Figure 3.5a suggest that $\Delta_s < 0$, which is completely unphysical: this corresponds to a situation when the correlation between two spins grows with the distance between the spins. The results in Figure 3.5b suggest that the Δ_ε value does not converge. In the flow starting from high temperature configurations, the temperature appears to remain high as the RBM flow length increases.

We also explore larger values of α equal to 4, 5 and 6. The coupling strengths for spins a distance of 2 units apart will have an interaction strength equal to $\frac{1}{2^\alpha}$. In the case of $\alpha = 3$ and $\alpha = 6$ we would have an interaction strength of $1/2^3 \approx 0.125$ and $1/2^6 \approx 0.016$ respectively. As α increases the interaction strength will approach zero for all interactions a distance of 2 or further apart. For the Ising model interactions further than 1 unit apart have a strength equal to zero. For the long range spin chain we therefore expect the critical temperature to decrease as α increases, and approach the critical temperature of the Ising model, $T_c \approx 2.269$. Using MCMC simulations we determine the critical temperatures and spin scaling dimensions as we did for $\alpha = 3$ by determining the temperature at which $\Delta_\varepsilon = 1$. The estimated temperatures and scaling dimensions for the different values of α are shown in Table 3.1. We do indeed see a decrease in T_c as α increases.

Table 3.1 MCMC estimates for T_c and Δ_s for various values of α for the long range spin chain.

α	Estimate of T_c	Δ_s
3	7.7	053
4	5.4	0.52
5	3.8	0.36
6	3.1	0.25

Table 3.2 Convergence values for Δ_s and Δ_ε for flows of length 1 up to 100 for different values of α. A value of NC denotes "no convergence" which describes values which do not show convergence over flow lengths of 1 to 100 (such a plot can be seen in Figure 3.5b).

	Flows start T	$\alpha = 3$ $\Delta_s = 0.53$ $T_c = 7.7$	$\alpha = 4$ $\Delta_s = 0.52$ $T_c = 5.4$	$\alpha = 5$ $\Delta_s = 0.36$ $T_c = 3.8$	$\alpha = 6$ $\Delta_s = 0.25$ $T_c = 3.1$
Δ_s	$T = 0$	0.175	0.15	1	NC
	$T = T_c$	0	0.05	0.05	0
	$T = 14$	-1.75	0	NC	-1.75
Δ_ε	$T = 0$	0.2	NC	NC	NC
	$T = T_c$	1.2	0.8	1.3	1.2
	$T = 14$	NC	1.1	NC	0.3

We train three additional RBM networks using data from the long range spin chain with α set to 4, 5 and 6 in the same way as for $\alpha = 3$. By increasing the value of α we move closer to the Ising model. By studying these networks with larger values of α, we can determine if the RBM flows show more similar behaviour to the Ising model [10, 11, 70] as α increases.

Repeating the numerical experiments for $\alpha = 3$, we use the trained RBM networks to generate flows of length 1 up to 100 and calculate values for Δ_s and Δ_ε at each flow length. The RBM flows do not correctly reproduce the spin and energy scaling dimensions. The results for flows starting from configurations at low temperature $T = 0$, the critical temperature and high temperature $T = 14$ are summarized in Table 3.2. In the case of $\alpha = 4$ we see convergence for the value of Δ_s to values well below 0.52 for flows starting from temperatures of 0 and $T_c \approx 5.4$. For Δ_ε we only see convergence when starting flows from $T = 14$ and $T_c \approx 5.4$ where we converge to values of 0.8 and 1.1 respectively. In the case of $\alpha = 5$ we see convergence for the value of Δ_s when starting the flow from temperatures of $T = 0$ and $T_c \approx 3.8$ where we converge to values of 1 and 0.05 respectively where we would expect a value of $\Delta_s = 0.36$. For Δ_ε we only see convergence when starting flows from $T_c \approx 3.8$ where we converge to a value of 1.3. In the case of $\alpha = 6$ we see convergence for the value of Δ_s to values of 0 and -1.75 for flow starting temperatures of $T_c \approx 3.1$ and $T = 14$ respectively. The value we expect here is $\Delta_s = 0.25$. For Δ_ε we only see convergence when starting flows from $T_c \approx 3.1$ and $T = 14$ where we converge to values of 1.2 and 0.3.

When starting flows from configurations at T_c we see convergence in both values of Δ_s and Δ_ε for all values of α. However these values do not correctly match the values expected for the various long range spin chain models. Here we again see the spin scaling dimensions

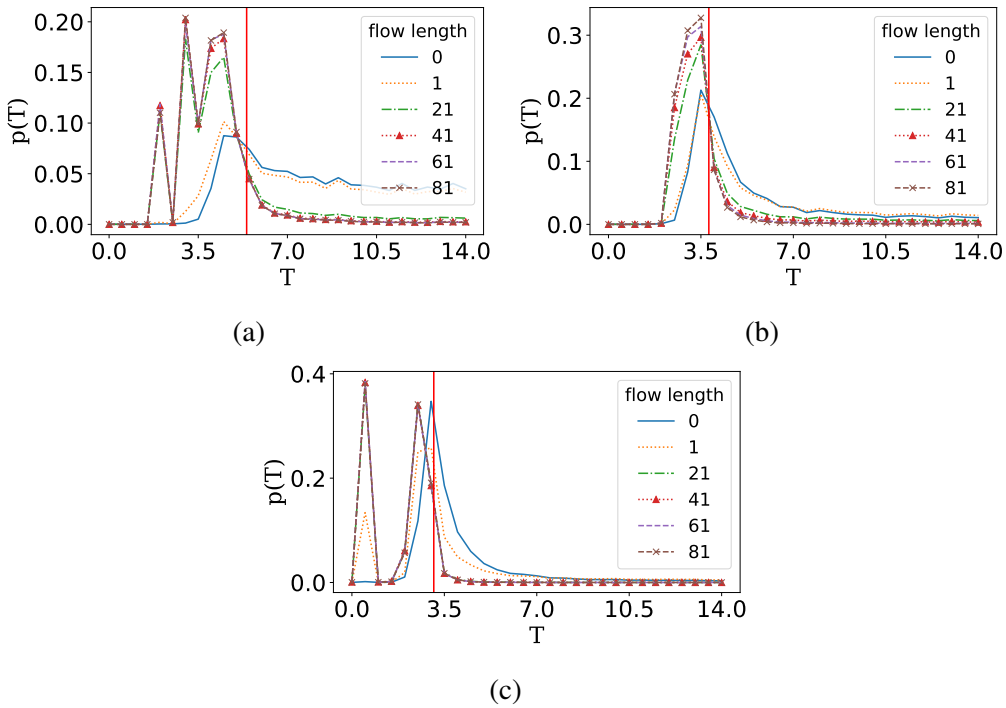


Fig. 3.6 Plots showing the average temperature of flows of different lengths for different values of α. (a) shows results for $\alpha = 4$, (b) shows results for $\alpha = 5$ and (c) shows results for $\alpha = 6$.

being underestimated as with $\alpha = 3$. In Figure 3.6 the average temperature of flows starting from T_c are shown for various flow lengths. If we compare these plots to the plot shown in Figure 3.4c we see that as α increases, the average temperature of flows moves closer to the vertical red line shown which gives the critical temperature for each respective model. The values of Δ_s at convergence (when starting flows from T_c) are all underestimates of the values determined by MCMC. This is in line with the average temperatures of flows depicted in Figure 3.6. For all values of α the temperatures converge to a temperature below T_c (to the left of the red line). As discussed earlier for the case of $\alpha = 3$, an underestimated spin scaling dimension signifies that the long distance correlations are over-estimated. When we have configurations at low temperatures there is a greater tendency for spins to align. This means that spins far away from each other will be more correlated at temperatures below T_c than at temperatures equal to T_c. The average temperatures shown in Figure 3.6 supports the fact that the spin scaling dimensions are underestimated.

In addition to the average temperature of flows, we can determine the percentage of flows at each flow length, that have a temperature near the critical temperature. We consider temperatures within 1 unit of the critical temperature to be near T_c i.e. $1 \leq |T - T_c|$. In Figure 3.7 the percentage of flow configurations near the critical temperature is shown. In Figure

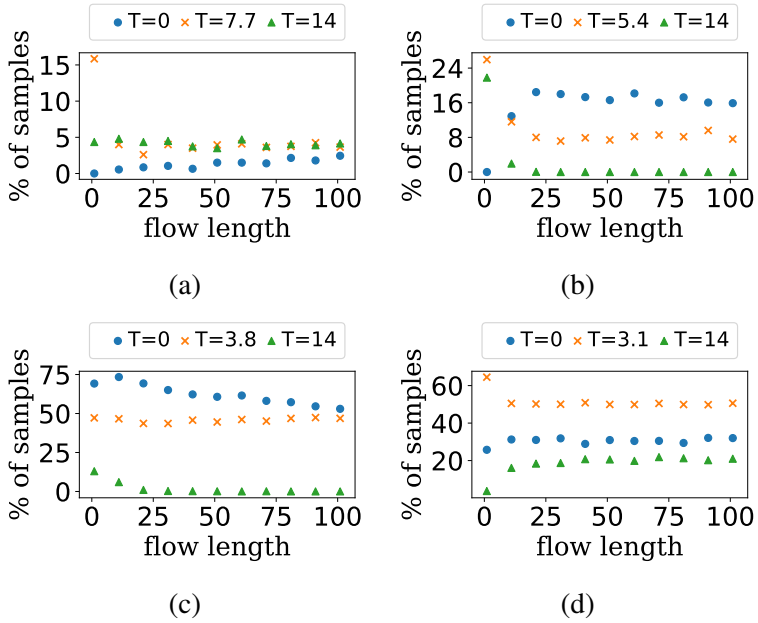


Fig. 3.7 Plots showing the percentage of samples that are near the critical temperature for different flow lengths and different values of α. (a) shows results for $\alpha = 3$, (b) shows results for $\alpha = 4$, (c) shows results for $\alpha = 5$ and (d) shows results for $\alpha = 6$

3.7a the percentage of flows at a flow length of 100 is below 5% regardless of the starting configurations temperature. When α is increased to a value of 6 as shown in Figure 3.7d the percentage of flows near $T_c \approx 3.1$ increases between 20% and 50% which is a definite increase from flows at $\alpha = 3$. As α increases the long range spin model moves closer to the Ising model as interactions other than nearest neighbour interactions tend to 0. From [10, 11, 70] we know that flows generated by RBMs trained on the Ising model do move towards the critical temperature. The increase in the percentage of flows near T_c as we move closer to the Ising model is therefore expected.

3.2.2 Stacked RBM and RG comparison

In this section we study the flow generated by successive stacked layers of a 3-layer deep RBM network. The first layer has $N_v = 64 \times 64 = 4096$ visible and $32 \times 32 = 1024$ hidden nodes, the second $32 \times 32 = 1024$ visible and $16 \times 16 = 256$ hidden nodes and the third $16 \times 16 = 256$ visible and $8 \times 8 = 64$ hidden nodes. The networks are trained in a greedy layer wise manner, i.e. each layer is trained separately and in order.

The first network is trained on 30000 configurations at a temperature of $T = 7.7 \approx T_c$ with lattice size 64×64 [28]. Once this network is trained, hidden configurations are generated by feeding training data to the first network. The hidden configurations generated by the first

RBM are the training data for the second RBM. Once trained the second RBM generates training data for the third and final layer.

To test if the stacked RBM flow resembles an RG flow, we compare to three steps of an RG flow. The deep network we have studied eliminates a quarter of the spins in each layer, which we mirror by block spinning groups of four spins. Consequently, the first step of RG will coarse grain lattices with $N_v = 64 \times 64 = 4096$ spins to lattices with $32 \times 32 = 1024$ spins, the second step then coarse grains to lattices with $16 \times 16 = 256$ spins and the third to lattices with $8 \times 8 = 64$ spins.

Our goal is to compare RG to unsupervised deep learning. One step of RG involves taking a linear combination of input (visible) spins to arrive at output (hidden) spins. For RG we perform an average of four spins to arrive at a new coarse grained spin in the output lattice. We can write the RG coarse graining as follows

$$h_{\mathbf{a}} = \sum_{\mathbf{i}} c_{\mathbf{a},\mathbf{i}} v_{\mathbf{i}}, \tag{3.19}$$

where $c_{\mathbf{a},\mathbf{i}}$ is a coefficient that ensures we perform a block spin average. Using this we can rewrite the $\langle vh \rangle$ correlators in terms of only the visible inputs

$$\begin{aligned} \langle v_{\mathbf{i}} h_{\mathbf{a}} \rangle =& \frac{1}{N_s} \sum_{n=1}^{N_s} v_{\mathbf{i}}^{(n)} h_{\mathbf{a}}^{(n)} \\ =& \frac{1}{N_s} \sum_{n=1}^{N_s} v_{\mathbf{i}}^{(n)} (\sum_{\mathbf{k}} c_{\mathbf{a},\mathbf{k}} v_{\mathbf{k}}^{(n)}), \end{aligned} \tag{3.20}$$

where n denotes the nth sample and N_s denotes the total number of samples. From this equation we can see that the $\langle v_{\mathbf{i}} h_{\mathbf{a}} \rangle$ correlator tells us information about the averaging performed as it depends on the coefficient $c_{\mathbf{a},\mathbf{k}}$.

An RBM consisting of one layer also takes a linear combination of input, visible nodes to get to an output, hidden node. The RBM accepts a vector of $N_v = L_v \times L_v$ sites and produces an output vector of size $N_h = L_h \times L_h$. The transformation performed by the RBM is

$$h_a = \tanh(\sum_i W_{ia} v_i + b_a^{(h)}), \tag{3.21}$$

which gives us the following correlator

$$\langle v_i h_a \rangle = \frac{1}{N_s} \sum_{n=1}^{N_s} v_i^{(n)} \left(\tanh(\sum_k W_{ka} v_k^{(n)} + b_a^{(h)}) \right). \tag{3.22}$$

In the case of the RBM transformation, all input spins can contribute to the average which produces an output spin. During training these weights are learned and some connections may be more important than others. It is evident from the above rewriting of $\langle vh \rangle$ in terms of only input visible spins, that this correlator captures important information about the coarse graining that occurs between inputs and outputs.

To develop a quantitative comparison of the flows of the stacked RBM and those of block spin RG, we calculate $\langle vh \rangle$ correlators between the initial spins populating the $64 \times 64 = 4096$ lattice and the spins populating each subsequent coarse grained lattice. The result is a matrix of values, $\langle v_i h_a \rangle$, indexed by a label i for a site of the original lattice and a label a for a site of a coarse grained lattice. Selecting a specific column, for example $\langle v_i h_1 \rangle$, gives the correlation of a specific coarse grained spin with all of the original spins. Rearranging the resulting $\langle v_i h_1 \rangle$ vector into a 64×64 matrix, produces a visual pattern of the correlation between the coarse grained spin and the original spins. In this way our results are used to produce a picture of how the coarse graining is achieved.

Figure 3.8 shows $\langle vh \rangle$ plots obtained from the RG flow. Plots (a) to (d) show typical correlation between the input spins and a block spin after one application of RG. The first four plots show a small highly correlated local patch with other areas having lower correlation values. The highly correlated patch corresponds to the set of spins averaged to produce the block spin, so that the coarse graining is indeed reflected in this correlator plot. Plots (e) to (h) show $\langle vh \rangle$ plots between the original spins and a block spin produced by two steps of RG. The plots clearly demonstrate that the patch of high correlation has increased in size, which is a consequence of the fact that after two steps of RG each block spin averages 16 of the original spins. Plots (i) to (l) show $\langle vh \rangle$ plots between the original spins and block spins produced by three steps of RG. The high correlation region has again increased, reflecting the fact that now each block spin summarizes 64 of the original spins. A significant feature of these plots is that the regions of correlation are local, which is a hallmark of the RG coarse graining in which short distance features are removed first by the coarse graining.

Figure 3.9 shows $\langle vh \rangle$ plots for the stacked RBM. Again here plots (a) to (d) show $\langle vh \rangle$ correlation between the training configurations (64×64 visible nodes) and hidden configurations (32×32 hidden nodes) from the first RBM, plots (e) to (h) show $\langle vh \rangle$ correlations between the training configurations (64×64 visible nodes) and the hidden configurations (16×16 hidden nodes) from the second RBM and plots (i) to (l) show $\langle vh \rangle$ correlations between configurations (64×64 visible nodes) of the training data and the hidden configurations (8×8 hidden nodes) from the third RBM. There is no clear difference between the plots for the first layer, second layer and third layer. There are some local patches but these are not as distinct as those of the RG flow of Figure 3.8. Comparison of figures 3.8 and 3.9

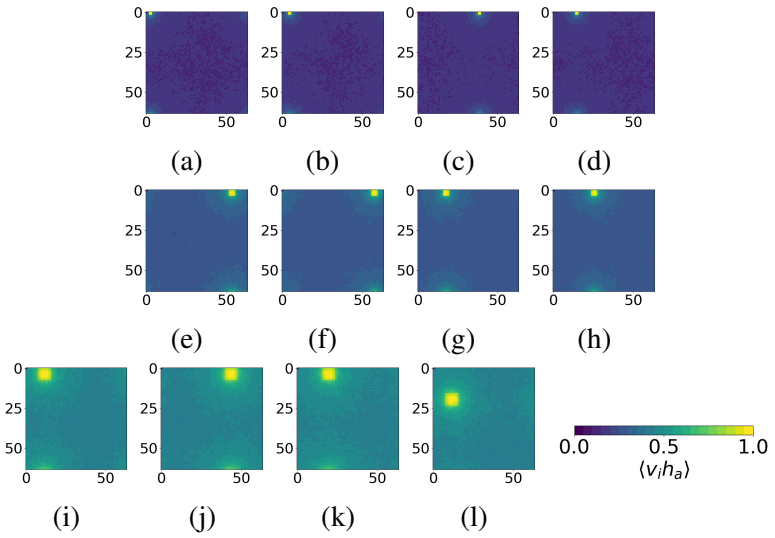


Fig. 3.8 RG $\langle vh \rangle$ correlation plot.
Plots (a), (b), (c) and (d) show typical $\left\langle v^{(1)} h^{(1)} \right\rangle$ plots where $v^{(1)}$ is an input spin and $h^{(1)}$ is a block spin produced by one step of RG.
Plots (e), (f), (g) and (h) show typical $\left\langle v^{(1)} h^{(2)} \right\rangle$ plots where $v^{(1)}$ is an input spin and $h^{(2)}$ is a block spin produced by two steps of RG.
Plots (a), (b), (c) and (d) show typical $\left\langle v^{(1)} h^{(3)} \right\rangle$ plots where $v^{(1)}$ is an input spin and $h^{(3)}$ is a block spin produced by three steps of RG.
In all 12 plots a single block spin's correlator with the complete collection of 64×64 input spins is plotted.

reveals that there is not much similarity, suggesting that the two coarse graining schemes are quite different. Another interesting aspect to explore is the temperature of each successive flow. Here we make use of the same supervised network used to measure temperature in Section 3.2.1. Figure 3.10 shows the average measured temperature of configurations at different steps in the RG and stacked RBM flows. After each step of RG, the temperature of the coarse-grained configurations appears to increase. After one application of RG we measure an average temperature of 6.25 which increases to 7 after two applications of RG and further increases to 7.5 after the third step of RG as seen in Figure 3.10b.

The stacked RBM appears to cause a decrease in temperature as we move through successive layers. The hidden vectors from the first RBM are at a high temperature with a maximum probability at a temperature of 12.5, the second and third RBMs have hidden vectors at a temperature just above 7.5.

Considering the comparison between $\langle vh \rangle$ plots and the temperature measurement of flows for the RG and stacked RBM network, we do not see convincing evidence that suggest the two are equivalent. In the case of the measured temperatures, we see very different

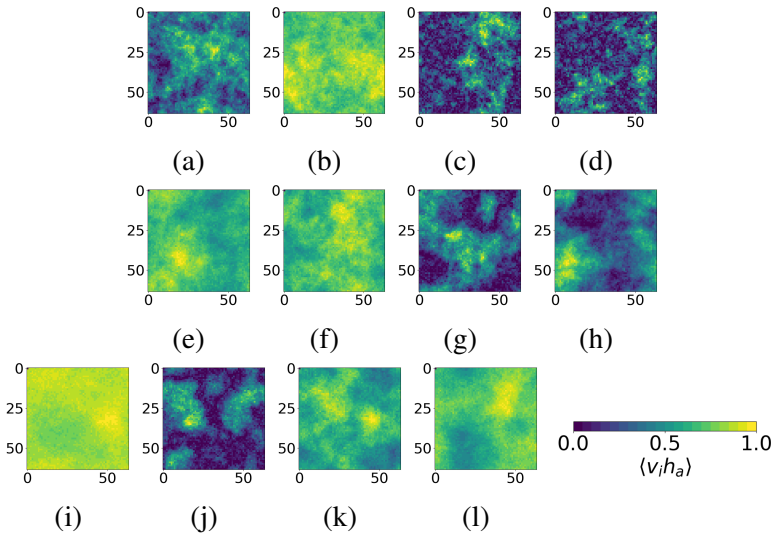


Fig. 3.9 RBM $\langle vh \rangle$ correlation plot. Plots (a), (b), (c) and (d) show a sample of the $\left\langle v^{(1)} h^{(1)} \right\rangle$ plots where $v^{(1)}$ is an input spin and $h^{(1)}$ is a hidden spin produced by the first RBM in the stacked network. Plots (e), (f), (g) and (h) show a sample of the $\left\langle v^{(1)} h^{(2)} \right\rangle$ plots where $v^{(1)}$ is an input spin and $h^{(2)}$ is a hidden spin produced by the second RBM in the stacked network. Plots (a), (b), (c) and (d) show a sample of the $\left\langle v^{(1)} h^{(3)} \right\rangle$ plots where $v^{(1)}$ is an input spin and $h^{(3)}$ is a hidden spin produced by the third RBM in the stacked network. In all 12 plots a single hidden node's correlator with the complete collection of 64×64 input spins is plotted.

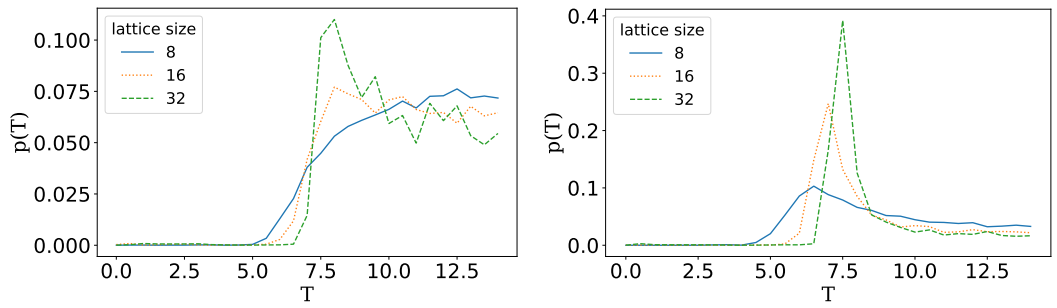


Fig. 3.10 Average probability of temperature versus temperature of a flow for a stacked RBM network trained on lattices at $T = 7.7$ and an RG flow of 3 steps. The RG flow and RBM flows are generated by applying both methods to the training set of 30000 lattices at $T = 7.7$ consisting of 64×64 spins. (a) shows results for the stacked RBM flow and (b) shows results for the RG flow.

behavior with the temperature flowing in opposite directions. With RG we see temperature

increasing with increased steps in the flow and with the stacked RBM we see a decrease in temperature with successive hidden vectors produced by the RBM layers.

For the two-dimensional Ising model as explored in [70], the flows went to a higher temperature in both the case of RG and that of the stacked RBM network. This contrasts with the flow of the long range spin model where we see an increase in temperature when applying successive steps of RG and a decrease in temperature when applying successive RBMs.

3.3 High temperature expansion

In this section we derive the high temperature expansion for the long range spin chain and explore the idea of performing an analogous expansion for the RBM. The high temperature expansion in statistical mechanics allows us to intuitively understand how certain quantities such as correlators behave at high temperature. This expansion can be performed when $\beta = 1/T$ is small ($\beta << 1$). To calculate the correlator $\langle X \rangle$ for a given spin model with Hamiltonian $H(s)$ we have

$$\langle X \rangle = \sum_s e^{-\beta H(s)} X. \tag{3.23}$$

To expand the above expression for high temperature we expand in powers of β. We know that each successive term in the expansion will get smaller as the nth term will be $\beta^n H^n/n!$. Here we have β^n getting smaller as n increases as well as $n!$ in the denominator. We begin by exploring the high temperature expansion for the long range spin chain.

3.3.1 Expanding the long range spin chain

The Hamiltonian of the long range spin chain, includes pair-wise interactions between every spin with an interaction strength of $1/n^\alpha$, where α is a constant and n is the distance between the pair of spins. When we perform a high temperature expansion for this model, the first order term of βH (where H is given in equation (3.1)) already contains a term with $s_{\mathbf{i}} s_{\mathbf{i+n}}$ which is scaled by $1/n^\alpha$. We know that $s_{\mathbf{i}}^2 = 1$ and $\sum_{\mathbf{s}} s_{\mathbf{i}} = 0$ because spins take values of ± 1. This means that when we expand in powers of β, only terms which have squared spins will survive. The two-point correlation function $\langle s_{\mathbf{i}} s_{\mathbf{i+n}} \rangle$ will therefore have the largest contribution from the first order term. We can thus write the two-point correlator as

$$\langle s_{\mathbf{i}} s_{\mathbf{i+n}} \rangle \propto \frac{\beta}{n^\alpha} = \beta n^{-\alpha}, \tag{3.24}$$

which demonstrates correlations have a power law fall off in the high temperature expansion. We now explore an analogous expansion for the case of the RBM.

3.3.2 Expanding the RBM for small E

The form of the RBM probability distribution is inspired by the Boltzmann distribution of statistical mechanics. We might therefore think that we could apply a high temperature expansion to the RBM as is done for spin systems. For the RBM there is no direct variable linked to the temperature of a system and it is not obvious what the notion of temperature means in this setting. If we consider expanding the RBM, we require some parameter to be small which we could expand in. Let us assume the energy to be small. If the energy is small we could expand the following two-point correlator $\langle v_i v_{i+n}\rangle$ as

$$\begin{aligned}\langle v_i v_{i+n}\rangle =& \frac{1}{Z}\sum_{\mathbf{v}}(\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})})v_i v_{i+n} \\ =& \frac{1}{Z}\sum_{\mathbf{v}} v_i v_{i+n}\sum_{\mathbf{h}}(1-E(\mathbf{v},\mathbf{h})+\frac{E^2(\mathbf{v},\mathbf{h})}{2}- \\ &-\frac{E^3(\mathbf{v},\mathbf{h})}{3!}+\dots),\end{aligned} \tag{3.25}$$

where the energy for the RBM is given in equation (B.1). The RBM's energy function is linear in the visible and hidden spins so the first order term will only have a $\mathbf{v}^T\mathbf{b}^{(\mathbf{v})}$ term remaining as this term is not removed by the sum over $\mathbf{h}$. This term however will not result in a non-zero term when calculating $\langle v_i v_{i+n}\rangle$ due to $\sum_{\mathbf{v}} v_i = 0$.

When we have even powers of the expansion, such as E^2, there will be terms $h_{\mathbf{i}}^2$ which when summed over $\mathbf{h}$ will be equal to 1. Using this we calculate the two-point correlator as

$$\begin{aligned}\langle v_i v_{i+n}\rangle =& \frac{1}{Z}\sum_{\mathbf{v}}\left(\sum_{\mathbf{h}}(1-E+\frac{E^2}{2}-\dots)v_i v_{i+n}\right) \\ =& \frac{1}{Z}\sum_{\mathbf{v}}\left(\sum_{\mathbf{h}}(0-0+\frac{E^2}{2}-\dots)v_i v_{i+n}\right) \\ =& \frac{2^{N_h}}{Z}\sum_{\mathbf{v}}\Big(v_i v_{i+n} b_i^{(v)} b_{i+n}^{(v)}+ \\ &+\sum_{j=1}^{N_h} v_i v_{i+n} W_{i,j} W_{i+n,j}\Big) v_i v_{i+n} \\ =& \frac{2^{N_h}}{Z}\left(b_i^{(v)} b_{i+n}^{(v)} + (\mathbf{W}\cdot\mathbf{W}^T)_{i,i+n}\right).\end{aligned} \tag{3.26}$$

where we denote the off diagonal components of the matrix $(\mathbf{W}\cdot\mathbf{W}^T)$ by $i, i+n$. For the long range models the one-point correlator is zero in the high temperature expansion. The energy function of the RBM has terms linear in $v_\mathbf{i}$ so the expectation of $\langle v_i\rangle$ would be non-zero if $b_i^{(v)}$ is non-zero. In order for us to match the one-point function between the RBM and spin models we require that all elements of $\mathbf{b}^{(\mathbf{v})}$ are zero. Assuming that the visible bias term is zero we now have the following expression for the two-point function

$$\langle v_i v_{i+n}\rangle = \frac{2^{N_h}}{Z}(\mathbf{W}\cdot\mathbf{W}^T)_{i,i+n}. \tag{3.27}$$

From the above analysis we know that $(\mathbf{W}\cdot\mathbf{W}^T)_{i,i+n}$ requires a power law fall off as n increases if the RBM is to correctly reproduce the one and two-point spin correlators of the long range spin chain.

We explore these analytic results below comparing the RBM correlators derived here to those derived for the long range spin chain. We do this using a RBMs trained on MCMC long range spin chain data at high temperature, $T = 14$, with values of 3, 4, 5 and 6 for α.

3.3.3 Numerical analyses of the high temperature expansion

In the high temperature expansion for the long range spin chain the two-point correlation function has a power law fall off. If we apply an expansion to the RBM for small energy, $\mathbf{W}\cdot\mathbf{W}^T$ should demonstrate this same behavior if we have correctly learnt the probability distribution of the spin model. To check this we train four RBMs on data from the long range spin chain model at high temperature, $T = 14$ with α set to 3, 4, 5 and 6. The training data is made up of 25000 samples for each network.

The input lattices used to train the RBM are rearranged from configurations of size $L_v \times L_v$ to vectors of length $N_v = L_v \times L_v$. When we look at a value $(\mathbf{W}\cdot\mathbf{W}^T)_{i,j}$ we must take into account which spin sites the visible nodes v_i and v_j relate to on the lattice of size $L_v \times L_v$. Spins v_i and v_{i+n} may appear to be a distance of n apart however if they are from a lattice of size $n \times n$ ($L_v = n$) they are a distance of 1 away from each other.

We determine the location $\mathbf{i_s}$ on the original lattice, of a visible spin v_i as

$$\mathbf{i_s} = (i \backslash L_v, i \bmod L_v), \tag{3.28}$$

where the symbol $\backslash$ denotes integer division (the truncation of decimals) and the mod operation defines taking the remainder of the division of i/L_v. This allows us to calculate distances between v_i and v_j according to locations on the spin lattice which we denote by $\mathbf{i_s}$ and $\mathbf{j_s}$.

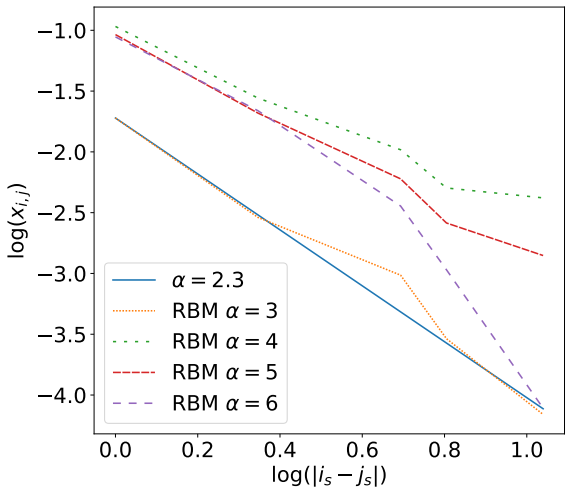


Fig. 3.11 The lines labelled RBM show the log of average values of off-diagonal components, $x_{i,i}$ versus the distance between spins v_i and v_j given by $|\mathbf{i_s} - \mathbf{j_s}|$. The weight matrices are from RBMs with $N_v = 10 \times 10$ and $N_h = 9 \times 9$ trained on long range spin chain data at $T = 14$ and the specified values of α.
The line labelled $\alpha = 2.3$ shows the log of a power law fall off of $1/|\mathbf{i_s} - \mathbf{j_s}|^{\alpha}$ with $\alpha = 2.3$.

The value $(\mathbf{W} \cdot \mathbf{W}^T)_{i,j}$ tells us about the connection between visible nodes v_i and v_j. We know that diagonal elements of $\mathbf{W} \cdot \mathbf{W}^T$ relate to connections between $v_i v_i$. We want to check if the connections between visible nodes v_i and v_j, where $i \neq j$, show a power law fall off as the distance $|\mathbf{i_s} - \mathbf{j_s}|$ increases.

Values of $(\mathbf{W} \cdot \mathbf{W}^T)_{i,j}$ are normalized by dividing by the diagonal component $(\mathbf{W} \cdot \mathbf{W}^T)_{i,i}$ to obtain $x_{i,j}$

$$x_{i,j} = \frac{(\mathbf{W} \cdot \mathbf{W}^T)_{i,j}}{(\mathbf{W} \cdot \mathbf{W}^T)_{i,i}}. \tag{3.29}$$

We plot the distances $|\mathbf{i_s} - \mathbf{j_s}|$ versus the average values of $x_{i,j}$ in Figure 3.11. In this plot values of $x_{i,j}$ for each of the four RBMs are shown by the dashed lines labelled "RBM". The solid line shows a power law fall off for a value of $\alpha = 2.3$ which has a similar gradient to the dashed lines shown. Here we see that there is a clear power law fall off of the two-point correlators learnt by the RBMs. The fall off of $\alpha = 2.3$ matches the fall off for the RBM trained on data with $\alpha = 3$. However for the values of α equal to 4, 5 and 6 we see similar gradients where we would have expected a steeper fall off. This suggests that the weight matrices of the RBMs have not correctly determined the exact power law fall off of the models they are trained on. As discussed in Section 3.3.1 the one-point function for the long range spin chain will be zero as there are no terms linear in $s_{\mathbf{i}}$ in the Hamiltonian. In the RBM energy function there are terms linear in v_i after summing over the hidden vectors. We

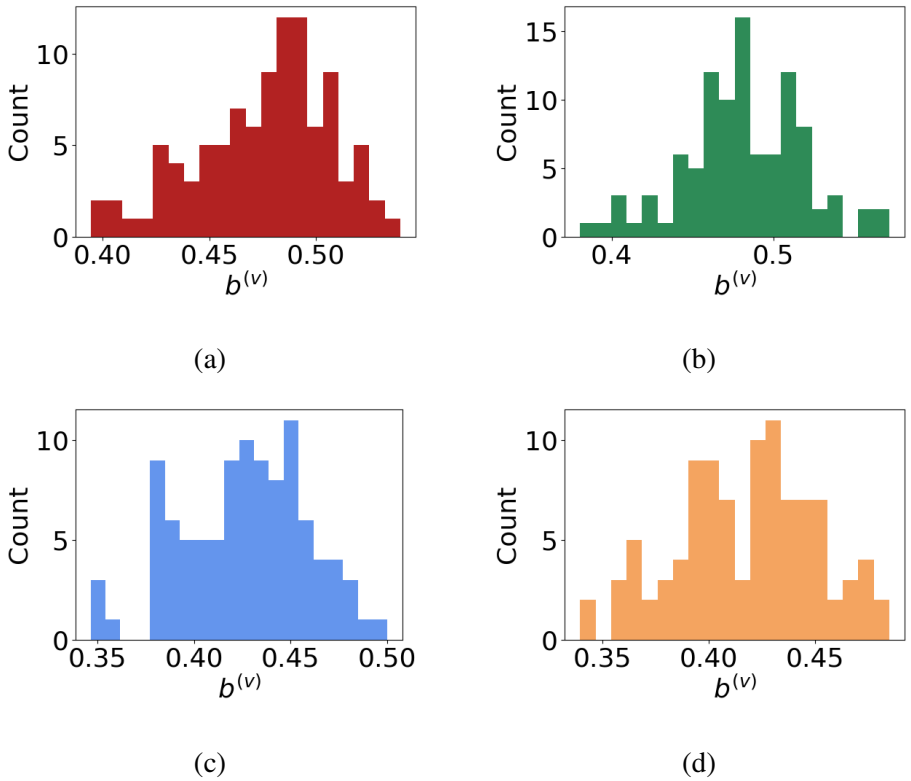


Fig. 3.12 Histograms showing the visible biases for RBMs trained on high temperature ($T = 14$) long range spin chain data with $N_v = 10 \times 10$ and $N_h = 9 \times 9$. Each plot shows results for a different value of α. (a) shows results for $\alpha = 3$, (b) shows results for $\alpha = 4$, (c) shows results for $\alpha = 5$ and (d) shows results for $\alpha = 6$.

thus require the values of the visible biases to be zero for the RBM to match the one-point function of the long range spin chain. Histograms showing values of $\mathbf{b}^{(\mathbf{v})}$ for the different values of α are given in Figure 3.12. The values lie between 0.35 and 0.6, which shows $\mathbf{b}^{(\mathbf{v})}$ is non-zero for all values of α. The RBM has therefore not correctly determined the one-point correlator of the long range spin models. Another interesting aspect to explore is the value of the energy of these four RBM networks and how this compares to models in statistical mechanics. In statistical mechanics we can have a system consisting of N lattice sites. The complete set of possible configurations consists of 2^N configurations. Only one configuration in this set will have all spins pointed up and only one will have all spins pointed down. Most configurations in the set will have roughly the same number of spins pointing up as pointing down. This gives a distribution for the energies which is highly peaked at specific values with almost no energies at other values. When considering values of the energy for the RBM shown in the histogram in Figure 3.13 we see that this is not the case. This distribution is not highly peaked at specific values as we would expect for a spin model

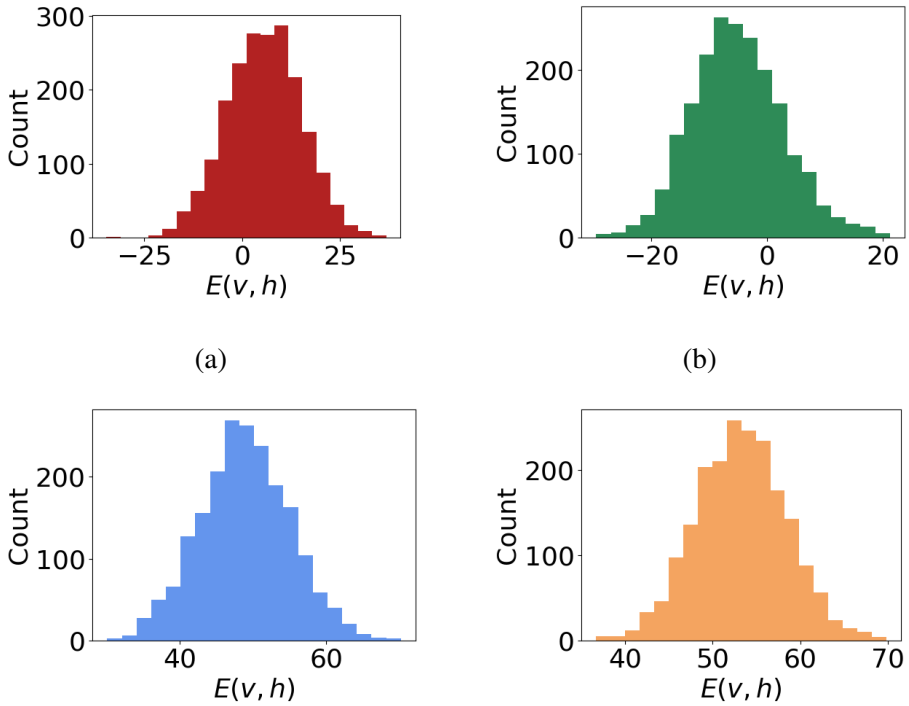


Fig. 3.13 Energy histograms for 2000 pairs of visible and hidden vectors for RBMs trained on long range spin chain data at $T = 14$ for various values of α. The RBM networks have $N_v = 10 \times 10$ and $N_h = 9 \times 9$. Each plot shows results for a different value of α. (a) shows results for $\alpha = 3$, (b) shows results for $\alpha = 4$, (c) shows results for $\alpha = 5$ and (d) shows results for $\alpha = 6$.

from statistical mechanics. This suggests that the RBM has some inherent differences from models in statistical physics even though it too uses the Boltzmann distribution.

3.4 Conclusions and discussion

We have explored the link between RG and unsupervised deep learning using RBM's trained on data taken from states of a long range spin lattice. This generalizes existing work [10, 11] exploring the two-dimensional Ising model with nearest neighbor interactions. Our results show important differences from those of [10, 11], which are discussed and interpreted in this section.

We have explored the RBM flow generated by the weights of a single RBM network, trained using spin configurations of a two-dimensional long range spin lattice. Data sets

generated at low temperature, at high temperature and near the critical temperature, have been considered. Our results demonstrate that regardless of the temperature from which the flow begins, the RBM flow fixed point does not resemble the critical point of the long range spin lattice. The scaling dimensions for the spin and energy operators do not converge to the values we expect at the critical point. In addition, when measuring the temperature of the flows, they do not converge to the critical temperature. This contrasts with results obtained for the Ising spin lattice where the Δ_s correlator is correctly reproduced and the flows appear to converge to the critical temperature [70].

Our study has also compared the RG flow to the flow produced by a stacked deep RBM network. Following proposals put forward in [70] this comparison was performed by comparing the $\langle vh \rangle$ correlator resulting from the two flows. $\langle vh \rangle$ plots from RG show local correlated patches surrounded by a large uncorrelated region. As the number of steps in the RG flow increases so does contrast between the patch of high correlation and the surrounding uncorrelated environment. This is very clearly a signature of the locality built into the block spinning RG procedure. Plots for the stacked RBM show random patches of correlation surrounded by dark uncorrelated regions. The patches lack the regularity of the patterns that emerge in the case of RG, and consequently they are not enhanced by subsequent layers of the deep RBM. As we move through successive layers of the RBM flow's $\langle vh \rangle$ plots there is no noteworthy organized change in the size or shape of the correlated patches which were observed for the RG flow.

In addition to the $\langle vh \rangle$ plots, the average temperature of successive steps in both the RG and the stacked RBM flows were determined. As expected the RG flow shows a clear increase in temperature with the length of the flow. This simply reflects the presence of relevant operators in the theory so that the fixed point is unstable. The RBM however shows a different behavior: the temperature decreases with the depth of the stacked network. For nearest neighbor Ising lattices temperature increases both along the RG flow and with the depth of the stacked RBM network, showing agreement on this very basic question. Even for this most basic question, the RBM flow and RG flow disagree for the long ranged spin lattice.

The lack of locality present in the $\langle vh \rangle$ RBM plots coupled with a decrease in temperature with successive applications of stacked RBM networks suggest that for the case of the long range spin chain, the RBM is not performing the same averaging as that present in an RG transformation. Locality is a signature of RG due to the manner in which the transformation is performed. If we study convolutional RBMs we know that locality is inherently built into the network [77]. With a standard RBM it is interesting to consider if it too performs a transformation with locality as a signature as this is not hard-coded into the structure of the network. If locality was present in the RBM averaging this would be convincing

evidence that these transformations are related. Even if we disregard locality as an important characteristic, the RBM shows important differences with the temperature flow through a stacked RBM network. The flow of temperature relates to relevant and irrelevant operators. In RG relevant operators grow and irrelevant operators die. If the RBM is performing an RG coarse graining, it should show this same behaviour with regards to the flow of relevant and irrelevant operators. Our analysis of the temperature flow suggests that this is not the case.

Our results clearly demonstrate that the coarse graining performed by the RBM does not correspond to block spin RG of the long range spin lattice. For the nearest neighbor Ising model there is at least some evidence that deep RBM's are reproducing some features of the RG flow. Why does the long ranged spin lattice differ so significantly from the nearest neighbor Ising lattice? In the long range spin lattice, interactions decrease smoothly in strength as the distance between the interacting spins increases. The net result is that more distant spins are correlated by interactions in the long range spin lattice, as compared to the Ising lattice whose interactions switch off abruptly as one moves beyond nearest neighbors. Consequently, it is harder for the RBM to recognize locality in the emergent patterns. Without locality, the resulting coarse graining is nothing like RG.

Setting aside this disagreement, there maybe a more fundamental obstacle to comparing the flows. The RBM energy function is simple and linear in v and h. Setting the coefficient of the term linear in v corresponds to fixing the single spin expectation value, i.e. the magnetization of the lattice. There are no further parameters in the RBM energy function that could be used to fix higher order correlators, such as two-point or three point spin correlators. This implies that not all relevant parameters in the theory have been fixed, so that it would be surprising if the RBM and RG flows terminate at the same point in the space of all possible lattice models. Small differences between the two will be magnified by the flow making the two look very different. Including higher order terms in the RBM energy function might allow us to use additional properties present in the input data, during training. This would allow the RBM to correctly learn essential physical characteristics of the long range spin lattice. Clearly our most credible conclusion is that more work is required to explore the connection that may exist between RG and unsupervised deep learning!

3.5 Author's contribution

The above work is published in Physical Review E and is a collaboration between the author, Anita de Mello Koch, Nicholas Kastanos and Ling Cheng.

The author had a major contribution to the above published article which included the ideas presented, design of experiments, analysis and interpretation of results and writing the manuscript.

Chapter 4

Why unsupervised deep networks generalize

Training a deep network involves applying an algorithm, which fixes the parameters of the network, using a training data set as input. After training is complete, the deep network is evaluated, by studying the trained network's performance on unseen test data. The difference between how the network performs on the test data and how it performs on unseen data defines a generalization error. Networks that perform as well on unseen data as they did on training data, have a small generalization error. On the other hand, networks that do well on training data but fail on unseen data, have a large generalization error.

Even with an incomplete understanding of how deep learning works, we have definite expectations for the size of the generalization error, based essentially on common sense. If the training data set is much smaller than the number of parameters in the network, training can fit the data perfectly, so that not only trends in the data but also errors and noise are captured during training. Intuitively we expect the generalization error is large, a penalty for fitting noise. If the training data set is much larger than the number of parameters, training forces the network to ignore noise in the data and only consistent and repeated trends will be learned. In this case we expect the generalization error is small. For the deep networks we discuss in this article, we have hundreds of millions of parameters trained using data sets with tens of thousands of elements. These are typical sizes of data sets and numbers of network parameters in practical deep learning applications. Clearly then, we are squarely in the regime of large generalization errors. Remarkably however, for typical deep learning applications, the generalization error is small. This begs the question: why do deep nets generalize?

The fact that such a basic puzzle remains open underscores that despite the remarkable success of deep learning applications, there is till a lot to learn about how deep learning

works. One does not need to prove generalization since it is well established in practice. However, an understanding of why networks generalize will shed light on what properties define well trained networks and in this way it will suggest how to find better architectures and more efficient training algorithms. The importance of the generalization question is appreciated and it has been studied for decades. Early attempts to explain generalization, suggested [78, 79] that deep networks that have flat minima in the training loss generalize well. Intuitively this is rather appealing: a potential that is truly flat in a given direction does not depend on the corresponding coordinate, and it is not a parameter of the network. With a flat enough potential, the effective number of network parameters may be small enough that the generalization puzzle is resolved. Numerical studies seem to support this idea, by demonstrating that sharp minima lead to higher generalization errors [80]. The "flatness" of the potential can be quantified in terms of the stability of the network's output, under addition of noise to the network's parameters [81]. By exploiting this noise stability phrasing of the problem, a number of generalization bounds were derived [82–87, 83, 88], but the bounds obtained are not yet tight enough to resolve the puzzle. Another notion of noise stability is the networks ability to reject noise injected at its input or internal nodes of the network [89]. Using this version of noise stability [13] proved generalization bounds that are almost strong enough to resolve the generalization puzzle. The insights of [13] strongly motivated the study presented in this chapter, so it is worth reviewing some of the key ideas.

The article [13] starts off by considering a fully connected layer, meaning that each input neuron is connected to every output neuron. The transformation from the input to the output neurons is linear and consequently one can perform a singular value decomposition of the weight matrix constructed by the training algorithm. Many of the resulting singular values are small, and can be set to zero. Consequently these should not be counted as parameters of the deep network, so that we have a much smaller number of "effective parameters". An operational test that there are a small number of singular values is captured by studying the noise rejection properties of the network: if noise is added to the input of a layer, it hardly has an effect on the output of that layer. The idea behind the equivalence of noise rejection and a small number of large singular values is rather simple: noise projects equally onto all singular vectors, but it is only the noise that is projected onto singular vectors associated to large singular values that is transmitted by the network. If only a small fraction of the singular values are large, only a small fraction of the noise is transmitted. For a network with many layers, the problem is non-linear so we can not use the singular value decomposition to analyze the network. However, the idea of noise rejection does generalize to a multi layer network, and this is why it is useful. For the multi layer case, we can compute a Jacobian which tells us how output changes in response to small changes in input. Trained multi layer

deep networks again exhibit noise rejection [13]. The observed noise rejection is evidence that the network only passes input information lying in a small subspace and so it again indicates that we are dealing with a very special map and that only a very small number of the total parameters that might appear in this map actually appear. This is a powerful idea: the naively very large number of parameters translates into a much smaller number of effective parameters and it is only the values of these effective parameters that actually determine the output of the deep net[1]. Using this much smaller number of effective parameters we come close to understanding why deep nets generalize.

The explanation provided in [13] is convincing, but it can only be part of the answer. To resolve the generalization puzzle, one needs an understanding of why so many of the singular values are small. Two natural questions to consider are: "Is it generic that such a large set of parameters ultimately get reduced to a very much smaller set of effective parameters?" and "What is the mechanism responsible for ensuring that almost all of the parameters of the deep network are irrelevant, that only a handful of the parameters actually count?" It is this class of questions that concern us in this study. A good answer should argue that the appearance of a large number of very small singular values is generic and further it should provide some insight into the basic mechanism for why this happens. Our approach to these questions shows that the structures that appear in the renormalization group descriptions of coarse graining in statistical physics and quantum field theory, naturally appear in unsupervised deep learning.

The renormalization group is used to generate a low distance effective theory (called the infrared or IR theory) starting from a microscopic theory (called the ultraviolet or UV theory). The UV theory can be specified in terms of a Hamiltonian, giving the energy of the system. Apriori, there are an enormous number of possible parameters of the theory corresponding to the possible terms that can be added to the Hamiltonian. Typically any term that is consistent with the symmetries of the theory can be added, so that the Hamiltonian is given by an infinite series and each term in the series has its own coupling constant. These coupling constants are the parameters of the UV theory. The renormalization group classifies parameters as relevant, marginal or irrelevant, according to how they evolve under the coarse graining. Relevant parameters grow as the coarse graining proceeds, irrelevant parameters decay and marginal parameters are fixed. Further, in situations in which the coarse graining has a well defined end point, i.e. in which there is a long distance effective description, most parameters of the problem are irrelevant. This provides a natural setting in which a problem with a potentially enormous number of parameters turns out to have very few effective parameters. Our central hypothesis in this chapter is that the mechanisms behind the renormalization group are also

[1]This reduction is dramatic. For the cases we study here, $\approx 0.3\%$ of the parameters are effective parameters.

at work in unsupervised deep learning, and that this leads to a resolution of the generalization puzzle.

Our motivation for this study was to map the unsupervised deep learning problem into the renormalization group language and thereby to shed light on how the generalization puzzle might be resolved. We will see that this is a useful exercise and what is more, it suggests a new framework within which a theoretical description of unsupervised deep learning can be developed. For the case of an RBM trained on Ising data, a study of the trained network proves that the network is coarse graining the input data, by discarding high momentum modes. The Ising example teaches us what to look for to recognize the renormalization group transformation, in the trained RBM. With this experience in hand, we find that autoencoders trained on various standard data sets all perform renormalization group transformations. This new point of view achieves a detailed understanding of what the network is doing and precisely what it learns, enabling us to give an algorithm to compute the network's parameters, given the learning data set. Although the resulting network does not perform quite as well as a trained network, it is an excellent initial condition for learning, cutting training times by a factor between 4 and 100 for the experiments we performed. Further, we are able to suggest a simple criterion to decide if a given problem can or can not be solved using a deep network.

In the next section we consider unsupervised learning using a restricted Boltzmann machine (RBM). To engineer an unsupervised deep learning setting, which is as similar to the renormalization group set up as possible, we train an RBM using data given by states of an Ising model. The trained weight matrix has a small number of large singular values and many small ones, which is the scenario of interest. We give the trained weight matrix a coarse graining interpretation, by showing that it is a quantitative match to block spin averaging, naturally used to implement the renormalization group for spin systems. A novel feature is that blocks overlap so that a spin in the UV lattice is averaged in more than one IR spin. This coarse graining interpretation gives an attractive interpretation of the singular value decomposition of the weight matrix. Singular vectors with a large singular value are relevant, and those with a small singular value are irrelevant. Further, the hidden singular vectors are obtained by discarding the high momentum modes of the visible singular vectors, so that the coarse graining being performed by the RBM is an exact match for the renormalization group transformation close to the free field fixed point! We go on to understand how the relevant degrees of freedom are determined, directly from the data set used for learning. We also consider a 3-layer stacked RBM, as well as autoencoders trained on the MNIST data set and data sets given by images of flowers and clouds. We present our conclusions in Section 4.2 and discuss the implications of thinking about unsupervised deep learning in

terms of a coarse graining perspective, instead of the more usual curve fitting paradigm. The Appendices collect technical results and further supporting data obtained from numerical experiment.

4.1 Unsupervised learning

In this section we study an RBM trained on data given by the states of an Ising model, at fixed temperature on a rectangular lattice. The renormalization group interpretation for the Ising problem is well studied, so this is a good starting point. Our primary goal is to explain why deep networks have far fewer parameters than suggested by naive estimates. The layers that are completely connected, i.e. they have a connection between every visible and hidden neuron, often contain the majority of parameters of the network. The RBM is a completely connected layer so it is a good example of a layer with many parameters.

In subsection 4.1.1 we study a single layer RBM trained on Ising data. This allows us to establish a link to the renormalization group. A useful way to implement the renormalization group is through block spin transformations, which are reviewed in Appendix B.2.2. We explore this connection both at the level of singular values and singular vectors. Singular vectors with dimension given by the number of input neurons will be called visible singular vectors, while singular vectors with dimension given by the number of output neurons will be called hidden singular vectors A network with three layers is also considered. One valid criticism of this work is that lessons learned from the Ising data might not be applicable to more usual applications. In subsection 4.1.2 we train using the MNIST data set, in subsection 4.1.3 we use a data set of images of flowers and in 4.1.4 one of clouds. These examples all exhibit the general features uncovered using the Ising data set.

4.1.1 Ising

We trained an RBM on Ising data. The input to the network is a set of 6400 dimensional vectors whose entries are the values (± 1) of spins located on the sites of an 80×80 rectangular lattice. The spin states used for training were generated using a Monte Carlo simulation with the Boltzmann distribution for the Ising Hamiltonian at a temperature of $T = 4$. The output from the network is a set of 1600 dimensional vectors whose entries are the values (± 1) of spins located on the sites of a 40×40 rectangular lattice. Thus, the weight matrix of the RBM has 10 240 000 entries, constituting the vast majority of the total number of parameters of the network.

During training we used batch sizes of 1000 samples, and training was performed with a total of 40 000 samples. The learning rate was 1×10^{-3} and we used a $\tanh(\cdot)$ activation function. The total number of training samples was chosen so that the training error had converged. See Appendix B.1 for a detailed description of the RBM including the equations we used in this study, and see Appendix B.4 for a discussion of the Ising model together with the Hamiltonian we used.

The trained weight matrix defines a linear map from input neurons to output neurons. One way to exhibit the structure of a linear map is by the singular value decomposition. In particular, it can interrogate how many parameters the linear map actually has. A large number of small singular values implies that there are not many parameters in the map. The weight matrix trained using Ising data has many small singular values, and the size of the singular values has a distinctive fall off, described by a smooth curve. The block spin averaging, also a linear operation, can also be studied using the singular value decomposition. As shown in Figure 4.1, we find that the plot of the singular values of the trained RBM weight matrix and the block spin coarse graining, are almost identical. Here we choose the size of the blocks that are averaged to produce a block spin, to maximize agreement with the RBM weight matrix. One novel feature of our results is that blocks that are averaged to produce the block spin, overlap. In the usual block spin averaging one averages a block of spins to produce an effective spin (as we do here) and the averaging rule does not allow blocks to overlap (unlike what we do here), i.e. each spin that is averaged contributes to only one effective spin[90]. There is however precedent for overlapping blocks: correlations between blocks, which are key for successful real space renormalization group algorithms[91], are captured with an overlapping block approach [92]. See Appendix B.5 for more details of the block spin averaging.

The real space renormalization group transformation is well understood: it is performing a coarse graining. The singular value decomposition provides a basis on which the coarse graining has a particularly simple action: coarse graining a visible singular vector gives the corresponding hidden singular vector multiplied by the singular value. Very small singular values correspond to visible singular vectors that are essentially coarse grained to zero. These are naturally associated with irrelevant degrees of freedom. The visible singular vectors that are associated with large singular values dominate the answer of coarse graining and hence are the relevant degrees of freedom. Consequently, the singular value decomposition is breaking the coarse graining up according to relevant and irrelevant degrees of freedom.

We know that the renormalization group is constructing the low energy effective theory. Consequently we expect the relevant degrees of freedom are associated with long distance behaviour of system. Another way to say this, is that the Fourier transform of the relevant

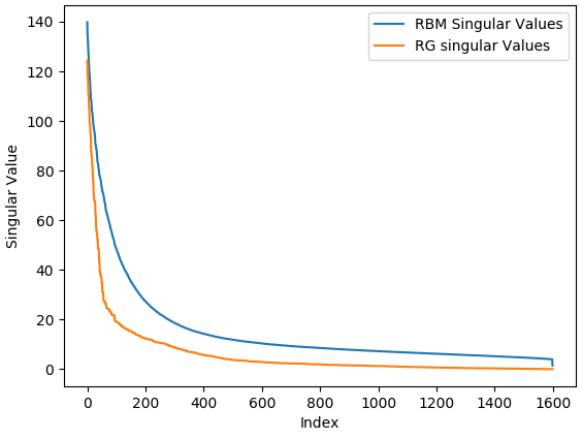


Fig. 4.1 A comparison of the singular values of the weight matrix of the trained RBM and the singular values of the matrix implementing the renormalization group by block spin averaging.

degrees of freedom should have its support in the lowest Fourier modes. We can explore the singular vectors to probe this expectation. The singular vectors are states on the lattice. The Fourier transform on the lattice is well defined, so we can take the Fourier transform of these states. To produce a one dimensional plot we average the magnitude of the FFT over angles to obtain a radial FFT. See Appendix B.6 for the details. This radial average is well motivated physically since we are simply exploiting the rotational symmetry of the problem. The rotational symmetry is broken by the lattice, so it is only an exact symmetry in the continuum limit.

The results for the Fourier transform of the singular vectors shown in Figure 4.2 and Figure 4.3 are intuitively appealing. Both the visible and hidden singular vectors associated with the large singular values have all of their support for small Fourier modes. Further, these hidden and visible singular vectors are identical, after a simple rescaling - multiplication by two. The hidden singular vectors are defined on a lattice with fewer total lattice sites, so that if we change units to keep the total length of the lattice fixed, the hidden singular vectors are obtained from the visible singular vectors by discarding the high momentum modes. As Figure 4.2 shows, this is not an approximate statement - after the high momentum modes of the visible singular vectors are discarded, there is an exact equality with the hidden vectors. This is remarkable because the procedure of discarding high momentum modes is an exact match to the form of the momentum space renormalization group transformation near the free field fixed point! The momentum space renormalization group transformation near the free field fixed point is reviewed in Appendix B.2.1. Since the transformation discards modes that are zero anyway, one might mistakenly conclude that the transformation is trivial. This

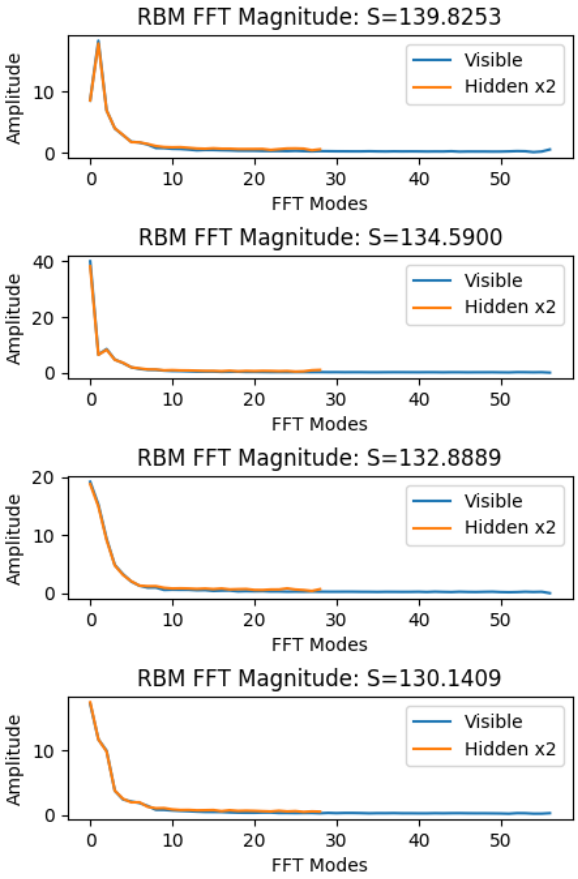


Fig. 4.2 The Fourier transform of the visible and hidden singular vectors associated to the five largest singular vectors, obtained from the singular value decomposition of the trained RBM weight matrix.

is not the case. Indeed, the Fourier transforms associated with the hidden and visible singular vectors are different, so that in the end the above transformation is extracting long distance features.

In contrast to this, the singular vectors associated with small singular values do not have most of their support at low Fourier modes, but are spread across the complete Fourier space. Further, the hidden and visible singular vectors are no longer related by the free field fixed point renormalization group. These modes are irrelevant and should simply be dropped. Dropping these does not affect the performance of the network.

The interpretation of the RBM in terms of coarse graining suggests that multiplying the weight matrix into a visible vector can be interpreted as follows: The size of the singular value tells you how relevant the mode is. We keep only the terms that are associated with large singular values. Each visible singular vector associated to a large singular value defines a relevant degree of freedom. The corresponding hidden singular vector is what the averaged

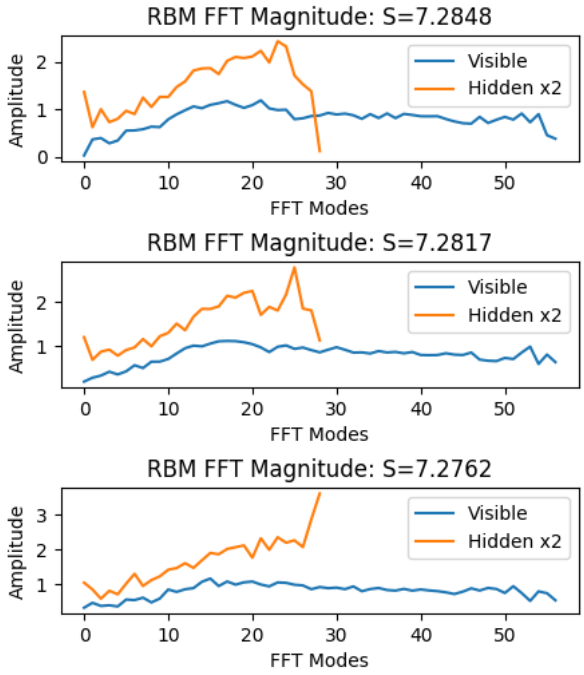


Fig. 4.3 The Fourier transform of the visible and hidden singular vectors associated to the five smallest singular vectors, obtained from the singular value decomposition of the trained RBM weight matrix.

relevant mode becomes after coarse graining. The visible singular vectors that correspond to small singular values correspond to irrelevant modes. The weight matrix decomposes the state it acts on into relevant and irrelevant modes and the irrelevant modes are discarded, while the relevant modes are averaged.

The specific form of the relevant modes is determined by the training data set. The fact that an effective description even exists strongly suggests that the data is concentrated on a sub manifold of the vector space in which the data lives. If this is the case, it is natural to expect that the visible singular vectors span this space. This simple picture provides a definition for the relevant modes, directly in terms of the data. A nice feature of this proposal, is that it is simple to test numerically using the training data. Using the data, we can evaluate the data covariance matrix and study it's eigen decomposition. If there are a few large eigenvalues and many small ones, the data is indeed concentrated on a sub manifold. The projection operator, projecting to this sub manifold is constructed using the eigenvectors associated to the large eigenvalues. Call this projector P_{data}; it's eigenvalues are, as for any projection operator, given by 0 or 1. We can also construct a projection operator that projects to the sub space spanned by the visible singular vectors associated to the large singular values.

Call this projector P_{trained}; it's eigenvalues are, again, 0 or 1. The operator

$$O = P_{\text{data}} P_{\text{trained}} P_{\text{data}} \tag{4.1}$$

will measure the alignment of these two subspaces. If all of O's eigenvalues are 0 or 1, and the number of eigenvalues equal to 1 matches the number of large singular values, then the subspaces are perfectly aligned. In general we expect many zero eigenvalues, plus eigenvalues that are close to, but not exactly equal to 1. The closer these eigenvalues are to 1 the better the two subspaces are aligned. The numerically computed eigenvalues, shown in Figure 4.4, demonstrate that there is a convincing alignment between the two subspaces. They are evaluated using the singular vectors associated to the largest 200 singular values and the eigenvectors associated to the largest 200 eigenvalues of the data covariance matrix. The numerical results show that of the 200 eigenvalues that might be 1, a majority of 177 of them are above 0.8 and only two are close to zero. This check is non-trivial: since both subspaces are embedded in a 1600 dimensional space, there was plenty of room for them to be completely orthogonal. These results are in agreement with observations reported in [93–95].

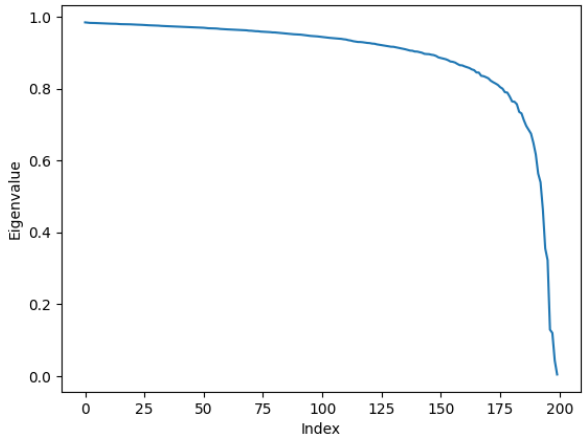


Fig. 4.4 The eigenvalues of $O = P_{\text{data}} P_{\text{trained}} P_{\text{data}}$ computed using the trained weight matrix and the Ising training data set.

Since we have developed an understanding of the irrelevant and relevant modes of the weight matrix it makes sense to revisit the counting of effective parameters. There are three important ideas

1. The sum over singular values can be truncated since many of the singular values are very small.

2. The visible singular vectors have support only at low Fourier modes. High Fourier modes are not parameters as they are set to zero.

3. Since the hidden singular vector is identical to the visible singular vector, we do not introduce new parameters for the hidden singular vector.

The reduction in the number of parameters implied by point 1 above is the reduction discussed in [13]. Keeping the largest 200 singular values (which amounts to dropping singular values less that 20% of the largest) reduces the number of parameters to 1 280 000, which is still very much larger than the number of data samples. For this particular example, the singular value curve is not very sharp and although this reduction in the number of effective parameters look impressive, its a little on the low side compared to typical. Even in more favorable cases, the reduction in the number of effective parameters achieved by the singular value truncation is not quite enough to resolve the generalization puzzle. Points 2 and 3 above are further reductions, not yet appreciated in the literature. This new reduction only becomes apparent after taking a Fourier transform, as instructed by the renormalization group logic. Looking at the singular vectors in Figure 4.2, it is clear that all modes above Fourier mode 10 vanish. The fact that we have taken a radial average implies that the number of distinct modes grows as the square of the mode number. Consequently, setting all modes above mode 10 to zero implies a further dramatic reduction in the number of effective parameters, by a factor of 36. This leaves us with 35 555 effective parameters, which is indeed less than the number of data samples in the training data set, explaining why we should expect the network to generalize i.e. why there is no over fitting. Thus, truncating the smallest singular values takes us a long way, but it is a nudge from the renormalization group that gets us over the line.

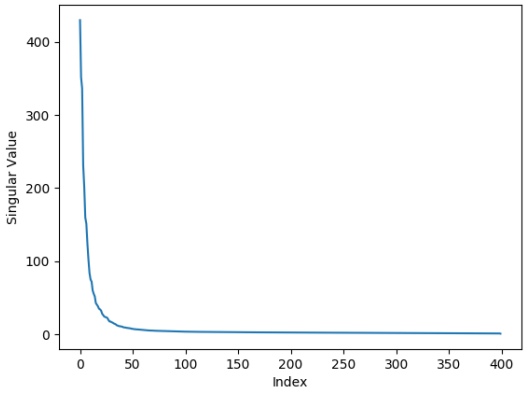


Fig. 4.5 The singular values obtained from a singular value decomposition of the trained weight matrix of the second layer of the stacked RBM.

An interesting extension of the study above is to consider stacked RBMs. We consider a three layer network. The first layer is as above, that is, the input is given by a set of states

of an 80×80 lattice and the output is a set of states of a 40×40 lattice; the second layer outputs states of a 20×20 lattice and the third states of a 10×10 lattice. Training uses the input data set described above. Training is carried out in a greedy layer-wise fashion.

The singular value curves are shown in Figure 4.5 and in Figure 4.6. It is clear that the singular value curves go to zero ever more steeply as the depth increases. Further, we have checked that the Fourier transform of the visible and hidden vectors are again in complete quantitative agreement, up to the scaling by 2 we found in the first layer. This is again a smoking gun for momentum space renormalization group coarse graining near the free field fixed point. The reader interested in these numerical results is referred to Appendix B.7. Consequently, our conclusions continue to hold when we increase the number of layers, to obtain a deep network. This is completely unsurprising, since the network is trained in a greedy layer-wise fashion.

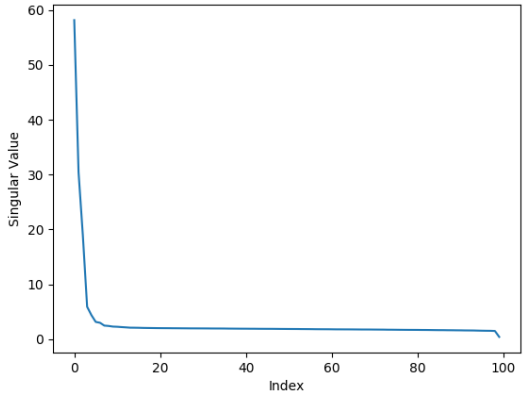


Fig. 4.6 The singular values from a singular value decomposition of the trained weight matrix of the third layer of the stacked RBM, trained using Ising data.

4.1.2 MNIST

A reasonable criticism of the previous section is that Ising data is rather special, and perhaps more closely related to usual renormalization group applications than the typical deep learning data set. For that reason, we consider the MNIST data set [96] in this section. The input to the RBM is images containing 28×28 pixels and the output from the RBM are images containing 14×14 pixels.

The singular values of the trained weight matrix are shown in Figure 4.7. There are again a large number of vanishing singular values, which may be set to zero. In the previous subsection, the strongest evidence, proving that the RBM is performing a renormalization group coarse graining, is given by verifying that the hidden singular vectors are given

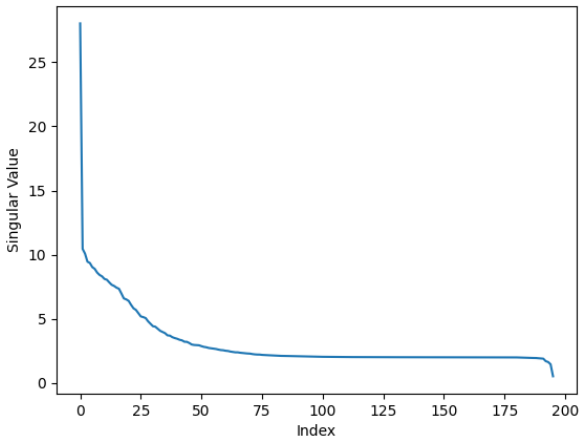


Fig. 4.7 The singular values obtained from a singular value decomposition of the trained weight matrix of an RBM trained using the MNIST data set.

by discarding the high momentum modes of the visible singular vectors. The radially averaged Fourier transform of the singular vectors obtained from the MNIST data set are shown in Figure 4.8, proving that the RBM is again carrying out a renormalization group transformation.

The link between the RBM and renormalization group coarse graining gives a rather detailed understanding of the parameters of the network. The key ideas are as follows

1. Visible singular vectors are given by the eigenvectors associated to the large eigenvalues of the data covariance matrix.

2. Hidden singular vectors are given by dropping the large Fourier modes of the visible singular vectors. One takes the Fourier transform of the visible singular vector using the Fourier transform defined by the lattice associated to the input data (a 28×28 lattice for MNIST), drops the higher Fourier modes and performs the inverse Fourier transform defined by the lattice associated with the output data (a 14×14 lattice for our RBM).

3. The singular values tell us whether modes are relevant, marginal or irrelevant. To estimate the singlar value associated to a visible singular vector, we perform a block spin coarse graining of the singular vector and then use the norm of this vector as an estimate for the singular value.

4. The biases tell us what visible and hidden singular vectors are favored by the RBM probability distribution. These should of course be related to the largest visible and hidden singular vectors. For numerical evidence supporting this identification, see

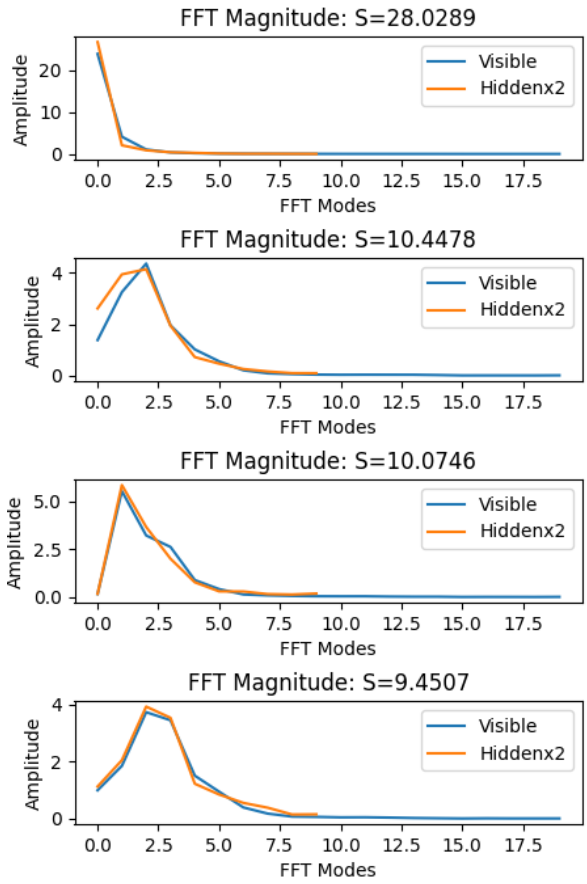


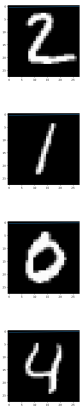
Fig. 4.8 The radial average of the Fourier transform of the singular vectors associated to the largest singular values, of the RBM weight matrix trained on the MNIST data set.

Figure 4.9. We take the visible (hidden) biases to be equal to the largest singular value, computed in 3. above, times the corresponding visible (hidden) singular vector.

These observations provide an algorithm to choose the parameters of the RBM using only the training data set. We call the resulting network an RGM for renormalization group machine. For a more detailed description of the RGM see Appendix B.3.

Table I shows a comparison of the RGM and the RBM. The RGM is not quite correct, although the images it produces are similar to images that are produced quite late in the training process. This suggests that the algorithmically computed weights are not far from the weights that are obtained from training. In this case, the RGM may well provide a good initial condition for training. The MNIST data set typically trains very quickly so it is not the best to use to test by how much training times are reduced. Nevertheless, we see that training from the RGM initial condition for 2 epochs gives roughly the same performance as an RBM trained using 8 epochs.

Table 4.1 Results of the RBM and RGM on the MNIST handwriting dataset. The RBM is trained for 8 epochs and the RGM has been trained for 2 epochs.

Original Image	RBM	RGM	Trained RGM
			

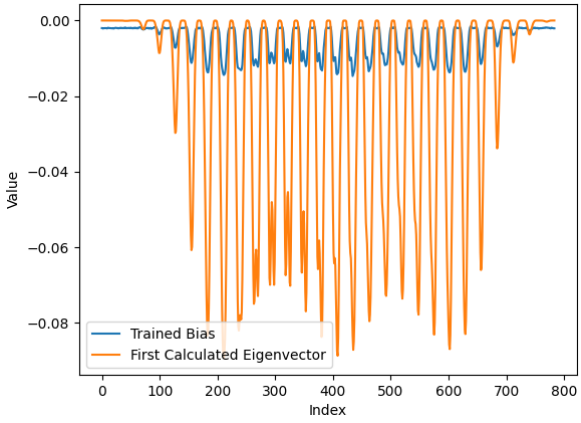


Fig. 4.9 A plot showing the visible bias and the visible singular vectors associated to the largest singular vector.

4.1.3 Flowers

In this section we have chosen to train an RBM on a flower data set, chosen specifically because a single layer RBM seems to have difficulty converging to a good set of weights when trained on this data. The TensorFlow Flowers data set [97] contains 3670 colour images divided into five classes which are often used for supervised learning. These images are less correlated to each other than the images in the MNIST data set, with complicated shapes appearing in some images, such as humans and teapots. We use the complete set of images to train a single layer RBM autoencoder. The images are all rescaled to 100×100, which is larger than the MNIST 28×28 input, and converted to grayscale before they are fed into

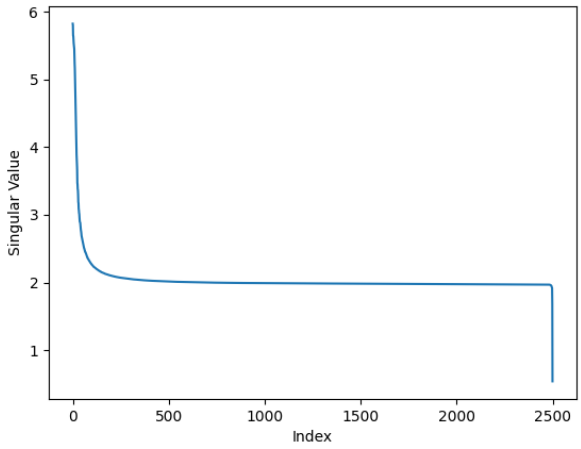


Fig. 4.10 The singular values obtained from a singular value decomposition of the trained weight matrix of an RBM trained using the TensforFlow flowers data set

the autoencoder. The autoencoder takes input data of size 100×100 and outputs a 50×50 image.

The singular value decomposition of the resulting trained weight matrix again supports the coarse graining interpretation of the RBM. From Figure 4.10 we again find a small number of large singular values, with a very sharp dropoff in size. In addition, from Figure 4.11 we see that the singular vectors asociated to large singular values again have all of their support at low Fourier modes. The hidden singular vectors are once again given by discarding the large Fourier modes of the visible singular vectors confirming the renormalization group link.

The results of the trained RBM (trained for 1000 epochs), the RGM and the trained RGM (trained for 10 epochs) are shown in Table 4.2. The images used to produce Table 4.2 did not appear in the training data set. The RBM used a Xavier initialization. For many of the images, it is clear that the RGM, whose parameters are agorithmically computed from the training data set, outperforms the RBM trained with 1000 epochs. This is compelling numerical support in favor of the understanding of the parameters of the RBM developed in this chapter. Further, if one now trains for 10 epochs, using the RGM as an initial condition, one obtains an autoencoder that outperforms the trained RBM.

It is worth noting that a single layer RBM trained with the same initialization and the same data set, but with images resized to 200×200 pixels, fails to converge. The block spin coarse graining rule is a very crude approximation to the true weight matrix. If the RBM is initialized to the block spin coarse graining rule, the RBM again converges to a valid autoencoder. For all of the experiments we conducted, initializing to the block spin coarse graining rule give significantly better results than the Xavier initialization, but not as good as

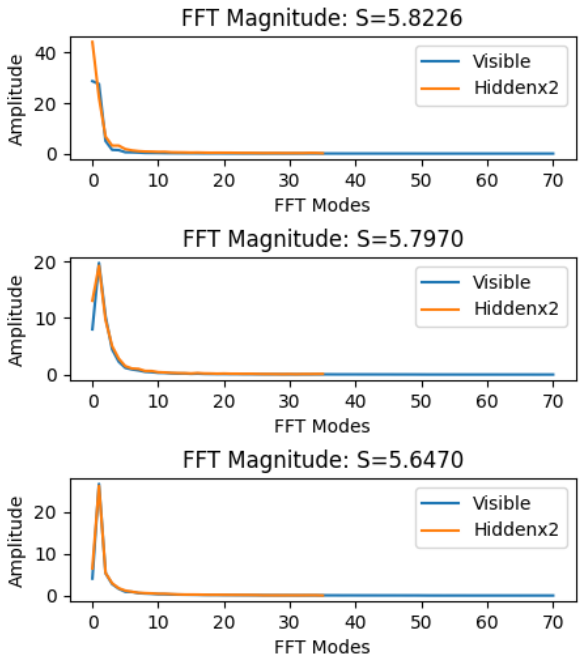


Fig. 4.11 The radial average of the Fourier transform of the singular vectors associated to the largest singular values of the RBM weight matrix trained on the TensorFlow flower data set.

initializing with the RGM. This again supports the idea that the RBM is performing a coarse graining.

4.1.4 Clouds

We used the Singapore Whole sky IMaging SEGmentation (SWIMSEG) data set [98] to train an RBM autoencoder. Images of clouds and the sky tend to be rather homogeneous and featureless, so this is another class of images with which to test our ideas. SWIMSEG is made up of 1013 sky images, each 600×600 pixels. To increase the number of images, we divide each image into 36 100×100 images and convert all images to grayscale. The autoencoder takes an input of 100×100 and outputs a 50×50 image. Training is initialized with a renormalization group block averaging initial condition. From Figure 4.12 and Figure 4.13 it is again clear that the trained RBM exhibits the hallmark features of renormalization group coarse graining.

4.2 Conclusions and discussion

A central conclusion of this study is that unsupervised deep learning is performing a coarse graining. More specifically, it is performing precisely the coarse graining of the momentum

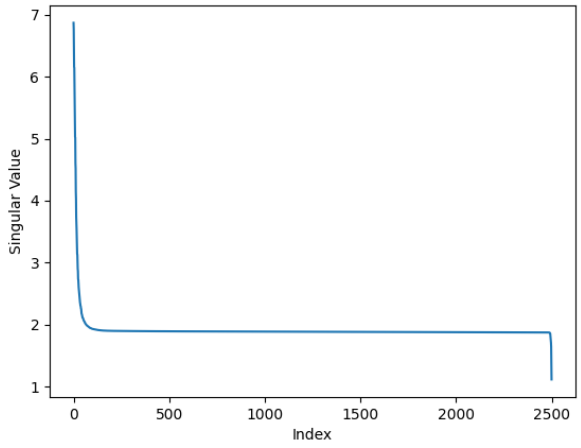


Fig. 4.12 The singular values obtained from a singular value decomposition of the trained weight matrix of an RBM trained using the SWIMSEG data set

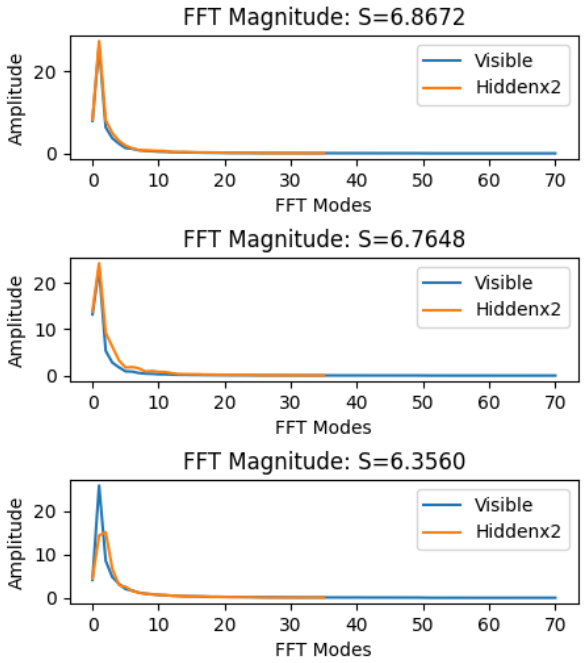


Fig. 4.13 The radial average of the Fourier transform of the singular vectors associated to the largest singular values of the RBM weight matrix trained on the SWIMSEG data set.

space renormalization group near the free field fixed point. The singular value decomposition replaces visible singular vectors with hidden singular vectors. The hidden singular vectors are obtained by dropping the large Fourier modes of the visible singular vectors - which is the definition of the momentum space renormalization group near the free field fixed point. Further, each replacement is associated with a singular value that determines when modes are relevant or irrelevant. The large singular values, defining the relevant modes, are associated

with singular vectors that have all of their support at low Fourier modes. These are precise mathematical statements that hold perfectly in every example we have presented.

The connection to coarse graining has implications for the generalization puzzle. First only the relevant modes given by the large singular values need to be retained, implying a dramatic reduction in the number of parameters, as pointed out in [13]. Second, relevant modes only have support at low Fourier modes, so that the coefficients of large momentum modes are set to zero. In addition, the coefficients appearing in the Fourier expansion of the hidden and visible singular vectors are not independent. This implies a further dramatic decrease in the number of parameters, which has not previously been pointed out in the literature and it is a direct consequence of the connection to the renormalization group.

The link to the renormalization group has allowed us to develop a rather detailed understanding of how the training data set determines the parameters of the trained network. This allows us to choose a set of network parameters given the training data set, defining what we have called a renormalization group machine or RGM. Although this choice is not perfect, it does give results that are close to a trained network and this choice provides a good initial condition that can speed up training. These results show that our study has shed light both on unsupervised deep learning and on the renormalization group. Indeed, the RGM shows how to define a renormalization group transformation for a finite data set.

Our study in this article is strongly motivated by the generalization puzzle. Deep networks seem to have so many parameters that over fitting is inevitable. Another facet of this puzzle is as follows: we break the data set into two disjoint subsets, one to be used for training the network and one for testing the trained network. There is nothing in the problem formulation that suggests that the results will be independent of how we perform the split into training and test data subsets. This independence, which is present in successful deep networks, is mysterious.

An equally vexing problem is the optimization puzzle: training a deep network boils down to finding the minimum of a loss function, which is a function of the millions of parameters of the deep network. We would generically expect that such a function, depending on such an enormous number of parameters, has many local minima and that a simple learning rule would almost always fail to take us to the global minimum. The different minima would correspond to different ways to thread a surface through the data. Typical deep learning problems, which do use simple training rules - typically some form of gradient descent implemented using back propagation - do manage to find great solutions and hence the puzzle.

A tacit assumption made by both puzzles is that deep learning is a form of curve fitting. In our opinion these puzzles suggest that curve fitting is the wrong paradigm to explain deep

learning. Our study suggests an alternative point of view: unsupervised deep learning is a sophisticated coarse graining which extracts trends and meaning from enormous data sets[2]. Consider an analogy which illustrates the idea. Imagine for the sake of example, that our system is a container filled with water. The state of the water is specified by an enormous list of positions and velocities of the water molecules, and the fine grained description uses the masses of the water molecules, as well as detailed and complicated interaction potentials. The coarse grained description uses a handful of parameters: things like the density of the fluid, its viscosity, the speed of sound in the fluid and its state is specified by a much smaller list of quantities like pressure and temperature of the water. Assuming the water is at equilibrium, we can choose any (large enough) subregion of the water (the training data) and use it to determine the parameters of the coarse grained description (the trained network). The rest of the system (test data) can then be used to test how good the description provided by the coarse grained theory is. Within the setting of coarse graining to obtain an effective description, it is natural to ask if the analysis depends on the details of the split into test and training data. Assuming that an effective theory exists and since the water is at equilibrium, we are certain that our results are independent of the details of this split, and hence generalization is a natural question and is guaranteed. There will obviously be fluctuations in the effective theory parameters, but these are small for large systems, so that to very good accuracy we get the same effective theory no matter how the training data is chosen. Thus, within the coarse graining framework, the generalization puzzle is a natural question and it is convincingly resolved.

Let us return to the optimization puzzle. By curve fitting logic, we expect there are many different minima, each corresponding to a different way of threading a surface through the data. It is inconceivable that these different surfaces give the same prediction on unseen data. On the other hand, coarse graining logic suggests that the network is finding the subspace on which the data is located. It is true that many of the parameters of networks trained on different training data sets will differ, because they will have different values for the irrelevant parameters. The point however, is that these networks will still agree on unseen data to good accuracy, because they have both learned the data subspace. This corresponds to a famous fact known as universality of the renormalization group flow: many different UV theories flow to the same IR theory.

The above discussion suggests a simple criterion to decide if a given problem can be solved or not. The criterion basically checks to see if the effective description constructed by coarse graining is well defined. One starts by selecting data samples at random, from the training data set, and computing their data covariance matrix. Once the number of large

[2]See [9, 10, 99, 100, 15, 16] where this idea was already considered.

eigenvalues of the data covariance matrix is no longer increasing as additional samples are selected, the process is terminated and the eigenvectors associated to the large eigenvalues are recorded. The process is then repeated, to produce a second set of eigenvectors associated to the large eigenvalues. If no matter how many times the process is repeated, the eigenvectors that are produced span the same subspace, the problem has a well defined effective description and it can be solved using a network. It would be interesting to explore the relation between this criterion and the one presented in [14].

Experience with the renormalization group suggests more. It relates a short distance, high energy theory to a long distance, low energy theory. According to our analogy, states of the high energy theory correspond to elements of the input data set, while the network outputs states of the coarse grained effective theory. The renormalization group splits parameters of the high energy theory into relevant, marginal and irrelevant parameters, according to how the coarse graining affects the value of the parameters. Relevant parameters grow under coarse graining, irrelevant parameters go to zero and marginal parameters maintain their size. The renormalization group explains why there are a handful of relevant or marginal parameters, but many irrelevant parameters. Changing the input data, for example by adding noise, changes the state of the high energy theory and hence it corresponds to changing parameters. Almost all changes of the input data correspond to changes in irrelevant couplings which will not change the low energy state, i.e. the output of the deep network. In this way the coarse graining framework predicts noise rejection of deep networks, which is established in practice.

There are a number of interesting directions that can be pursued. In what follows we list some of them.

- In this article we have focused on unsupervised learning. It would be fascinating to explore supervised learning in detail, in the coarse graining framework. If this framework is applicable, a key question is how the labels of the data are used to determine the relevant modes of the coarse grained description. There are also a number of simple tests that can be performed to test whether or not the coarse grained description is the correct one. For example, does one still see a large number of small singular values for the weights of fully connected layers? Further, if the weight matrix is initialized correctly, does one still find that the singular vectors associated to large singular values have their support on low Fourier modes? Are the hidden singular vectors still obtained from the visible singular vectors by discarding large Fourier modes?

- Our study considers learning data sets composed of images. The pixels are organized in a two dimensional lattice. This is an important ingredient: for the Fourier transform

the data needs to be organized in a sensible two dimensional lattice. For more general applications there might be no obvious way to define the Fourier transform. This is an interesting direction that deserves to be explored.

- The role of depth of the network can be studied in more detail. For stacked RBMs, where the deep network is trained layer-wise, we have seen that each layer is performing a renormalization group transformation. Thus, a stacked network is performing a renormalization group flow. Renormalization group flows can have a rich behavior, with the relevance and irrelevance of modes changing as the flow proceeds. It would be interesting to see if similar phenomena appear in a deep network.

- Our choice for the network parameters, in the form of the RGM, is close to what is produced by training. By improving our understanding of what determines the singular values of the trained weight matrix, it may be possible to further improve this choice. Giving an algorithm for the network parameters in terms of the training data, which is as good as parameters learned from training maybe too ambitious. What is however clear is that progress along this direction may lead to initial conditions that significantly reduce training times.

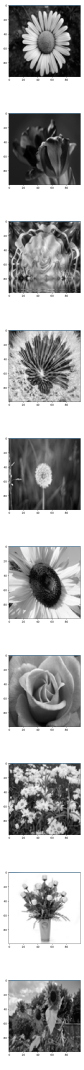
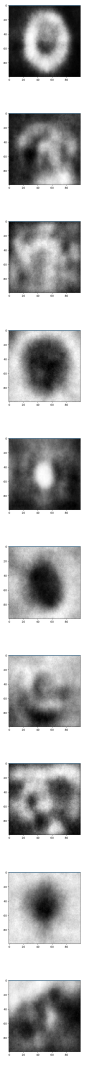
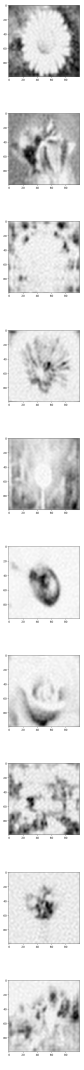
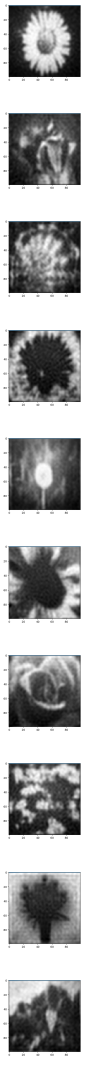
This is a rather incomplete list of questions. The coarse graining framework suggests many more. We look forwards to exploring these in the future.

4.3 Author's contribution

The above work has been submitted for publication. This work is a collaboration between the author, Robert de Mello Koch and Anita de Mello Koch.

The author had a major contribution to the above published article which includes the development and implementation of numerical experiments and contributions to writing the manuscript.

Table 4.2 Results of the RBM and RGM on the flower dataset. The RBM is trained for 1000 epochs and the RGM has been trained for 10 epochs.

Original Image	RBM	RGM	Trained RGM
			

Chapter 5

Discussion and conclusion

The key question addressed by this thesis is whether there is a link between RG and unsupervised deep learning. Although previous studies had suggested a link and provided circumstantial evidence for such a connection, the study presented in this thesis is the first to give compelling numerical evidence establishing the connection in a number of definite settings. The link provides a compelling starting point from which to develop an understanding of deep learning, as RG provides a rich set of ideas and is a well developed theory. This thesis develops techniques and methods to probe the link between RG and unsupervised deep learning. The results presented culminate in showing that a clear link does exist between the two coarse graining techniques. We focus on comparing a discrete version of RG, block-spin averaging developed by Kadanoff, to RBM networks and autoencoders in various settings.

This thesis develops techniques and methods to expose and quantify the link between RG and unsupervised deep learning. This logic culminates in quantitative results that establish the equivalence of the coarse graining performed by the trained RBM and coarse graining performed by RG. A central insight of this study is to focus on a discrete version of RG known as block-spin averaging, to establish its identity with trained RBM networks and autoencoders in various settings. A key technique that plays a central role is the singular value decomposition, which can be applied to both the block-spin averaging and to the trained RBM. The results obtained can be used to achieve an understanding of the values of trained parameters of the RBM and this is translated into more efficient training protocols. These results also shed light on deep theoretical issues in the field, including the generalization puzzle.

RG defines certain operators which are relevant, irrelevant and marginal. With successive applications of RG only the relevant and marginal operators remain in the new coarse grained description. If we can determine what parameters in a model are important for learning relevant and marginal operators we might be able to reduce the number of parameters tuned

during training. This would have a number of significant consequences such as reduced training times, reduced network size and novel architectures.

The data used in our first two studies consists of physical models of magnets. These models are made up of discrete spins which take values of ± 1. The first model studied is the Ising model where interactions are allowed only between nearest neighbor spins. The Ising model is well studied in physics and so has a rich set of analytically derived statistical properties which allow us to probe the RG and RBM coarse grainings. The second spin system we explore is the long range spin chain where interactions are allowed between any two spins on the lattice with an interaction strength proportional to the inverse distance.

Spin systems are a useful setting in which to explore this link as they have inherent locality as is the case in images. Nearby spins are highly correlated to one another. As the distance between two spins increases, their correlation decreases. This is true of images too as nearby pixels are more likely to have a similar colour than pixels which are far away from each other.

We generate data using Monte Carlo simulations and study both stacked and single layered RBM networks.

To compare RBM networks to RG we develop statistical techniques which allow us to study how the inputs and outputs are correlated. RG is characterized by locality. We introduce a technique to study the spatial correlators of the input and output correlators of each coarse graining method. This is a quantitative way to compare how the hidden/output nodes encode the visible/input data.

The final result presented here demonstrates that RBMs and autoencoders are performing a coarse graining equivalent to RG. This analysis is performed on magnetic spin systems in addition to more relevant datasets of MNIST, TensorFlow Flowers and SWIMSEG [96–98]. This result also shows that initializing weights to values related to the training data covariance matrix gives speed ups of training time of between 4 and 100.

Below each chapter is discussed, highlighting the major contributions of each.

5.1 Discussion of main results

Chapter 2 explores the extent to which the RBM flow converges to configurations at the critical temperature. The critical temperature of the Ising model is where a phase change occurs. At this point the spin system moves between a state of magnetization and a state of disorder where even nearby spins have little correlation. At the critical temperature we see a balance in the competing effects of minimizing energy versus maximizing entropy. A key finding is that the RBM produces configurations which exhibit the correct spin scaling

dimension but fail to reproduce the longer distance energy scaling dimension. The critical temperature is an unstable fixed point of the RG flow. This means that the temperature must be tuned very carefully if one is to flow towards this fixed point. The RBM flow is not the correct flow to consider as we have no depth which is a key characteristic of unsupervised deep learning.

In addition to exploring properties of the RBM flow we also study properties of stacked RBM networks consisting of 3 layers. We calculate correlations between the input and outputs of each coarse graining method at each step of the coarse graining. We then quantify these using spatial correlations. Both in the case of RG and the RBM we find that locality is present. This is evident in the fall off of the correlations as the distance between sites is increased. This is an encouraging result as locality is a key characteristic present in RG. In addition to the locality we also see an increase in the temperature as we move through layers of the RBM. This too is characteristic of the RG flow as there are relevant operators present in the Ising model which grow as we apply successive steps. These relevant operators increase the temperature as the critical temperature is unstable and so we flow to a stable fixed point at high temperature.

In chapter 3 we study the long range spin chain. In the case of the Ising model we find that the locality signatures and the temperature flow show similarities between RG and stacked RBMs. The Ising model only allows interactions between neighboring spins. In the long range spin chain we allow interactions between all spins proportional to the inverse distance between them. We study RBM flows of a single RBM network starting from temperatures below, above and at the critical temperature. Regardless of the starting temperature we find that the RBM flows do not flow towards the critical temperature and as a result the scaling dimensions are not correctly determined.

In addition to the RBM flow we again study stacked networks. In the setting of the long range spin chain we find that the flow of temperature for the RBM does not match that of RG. RG shows an increase in temperature with successive coarse graining steps. The RBM shows a decrease in temperature with successive steps. In addition to the discrepancy in temperature flow there is also a lack of locality present in the input output correlation plots. These results show that for the case of the long range spin chain, RBMs are not performing the same averaging as that of RG. This is an important difference to highlight when compared to the case of the Ising model. This suggests that when interactions are strictly local the RBM behaves in a similar way to RG.

In Chapter 4, a concrete result for why deep networks generalize is presented. We give convincing evidence that what RBMs and autoencoders learn are in fact representations equivalent to what is given by RG theory. We observe that the hidden and visible vectors are

identical in the low frequency modes and that high frequency modes have been discarded. By initializing weights to the low frequency modes of the data covariance matrix we see dramatic speed ups in training times of between 4 and 100. Understanding unsupervised deep learning from a curve fitting framework is the incorrect approach. This result shows that unsupervised deep learning is performing a coarse graining related to RG. Phrasing this problem in terms of coarse graining removes the paradoxes present in the curve fitting setting.

This work holds weight in the broader machine learning community as we demonstrate the low Fourier mode properties of the weight matrix and training speed ups in settings other than Ising. We study RBMs and autoencoders trained on the MNIST, a sky and cloud dataset (SWIMSEG) and TensorFlow's flower dataset. We show that the number of parameters which require training is reduced to a value below that of the number of training samples. This is due to the fact that many of the weight matrix's singular values are small and due to the relation between the hidden and visible singular vectors.

This result is a significant step forward in developing a theory for deep learning.

5.2 Future work and conclusion

In this work novel methods exploring the link between unsupervised deep learning and RG have been developed. More importantly this work culminates in the result that unsupervised networks are performing an RG coarse graining.

Further work includes studying supervised networks in an attempt to show whether they too are performing an RG like coarse graining. We could perform similar analysis to that presented in Chapter 4 on the trained weights.

In a supervised setting we care about the labels of each sample in the training data. If we are to view this in terms of coarse graining one such question to ask would be how the subspaces of each set of labelled samples differ from one another. This would give a clear method for distinguishing one labelled set from another.

In addition we make use of a Fourier transform to determine which modes in the training data are relevant. This relies on the data being organized sensibly on a two dimensional lattice. If this method is to apply to more general applications where the training data is not organized in this way, we may need to find a different criteria. This would be an interesting avenue to pursue further.

The RGM we develop shows similar characteristics to the fully trained weight matrix and reduces training times significantly. This guess could be improved by understanding better

what determines the singular values of the trained weight matrix. An improvement in this guess could further improve reductions in training times.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," Nature, vol. 529, pp. 484–503, 2016. [Online]. Available: http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, 2013.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[6] S.-C. Lin, W. F. Punch, and E. D. Goodman, "Coarse-grain parallel genetic algorithms: Categorization and new approach," in Proceedings of 1994 6th IEEE Symposium on Parallel and Distributed Processing. IEEE, 1994, pp. 28–37.

[7] J. Guo and M. Potkonjak, "Coarse-grained learning-based dynamic voltage frequency scaling for video decoding," in 2016 26th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS). IEEE, 2016, pp. 84–91.

[8] X. Peng, J. Hoffman, X. Y. Stella, and K. Saenko, "Fine-to-coarse knowledge transfer for low-res image classification," in 2016 IEEE International Conference on Image Processing (ICIP). IEEE, 2016, pp. 3683–3687.

[9] P. Mehta and D. J. Schwab, "An exact mapping between the variational renormalization group and deep learning," arXiv:1410.3831, Oct. 2014. [Online]. Available: http://arxiv.org/abs/1410.3831

[10] S. Iso, S. Shiba, and S. Yokoo, "Scale-invariant feature extraction of neural network and renormalization group flow," Physical review E, vol. 97, no. 5, p. 053304, 2018.

[11] S. S. Funai and D. Giataganas, "Thermodynamics and Feature Extraction by Machine Learning," Oct. 2018, arXiv:1810.08179. [Online]. Available: http://arxiv.org/abs/1810.08179

[12] E. de Mello Koch, "Dl and rg," https://github.com/ellendmk/DL_and_RG, 2020.

[13] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang, "Stronger generalization bounds for deep nets via a compression approach," arXiv preprint arXiv:1802.05296, 2018.

[14] S. Arora, S. S. Du, W. Hu, Z. Li, and R. Wang, "Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks," arXiv preprint arXiv:1901.08584, 2019.

[15] E. d. M. Koch, R. d. M. Koch, and L. Cheng, "Is deep learning a renormalization group flow?" IEEE Access, vol. 8, pp. 106 487–106 505, 2020.

[16] E. de Mello Koch, A. de Mello Koch, N. Kastanos, and L. Cheng, "Short-sighted deep learning," Physical Review E, vol. 102, no. 1, p. 013307, 2020.

[17] A. de Mello Koch, E. de Mello Koch, and R. de Mello Koch, "Why unsupervised deep networks generalize," arXiv e-prints, pp. arXiv–2012, 2020.

[18] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," Science, vol. 349, no. 6245, pp. 255–260, 2015.

[19] L. Deng, D. Yu et al., "Deep learning: methods and applications," Foundations and Trends® in Signal Processing, vol. 7, no. 3–4, pp. 197–387, 2014.

[20] Y. Bengio et al., "Learning deep architectures for ai," Foundations and trends® in Machine Learning, vol. 2, no. 1, pp. 1–127, 2009.

[21] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," science, vol. 313, no. 5786, pp. 504–507, 2006.

[22] N. Le Roux and Y. Bengio, "Deep belief networks are compact universal approximators," Neural computation, vol. 22, no. 8, pp. 2192–2207, 2010.

[23] ——, "Representational power of restricted boltzmann machines and deep belief networks," Neural computation, vol. 20, no. 6, pp. 1631–1649, 2008.

[24] Y. Bengio, Y. LeCun et al., "Scaling learning algorithms towards ai," Large-scale kernel machines, vol. 34, no. 5, pp. 1–41, 2007.

[25] A. Paul and S. Venkatasubramanian, "Why does deep learning work? - a perspective from group theory," arXiv:1412.6621, Dec. 2014. [Online]. Available: http://arxiv.org/abs/1412.6621

[26] Y. Bengio, A. C. Courville, and P. Vincent, "Unsupervised feature learning and deep learning: A review and new perspectives," CoRR, abs/1206.5538, vol. 1, p. 2012, 2012.

[27] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," arXiv:1611.03530, nov 2016. [Online]. Available: http://arxiv.org/abs/1611.03530

[28] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in Advances in neural information processing systems, 2007, pp. 153–160.

[29] H. W. Lin, M. Tegmark, and D. Rolnick, "Why does deep and cheap learning work so well?" J. Stat. Phys., vol. 168, no. 6, pp. 1223–1247, sep 2017. [Online]. Available: http://arxiv.org/abs/1608.08225

[30] J. Kogut and K. Wilson, "The renormalization group and the ε expansion," Phys. Rep. C, vol. 12, no. 2, pp. 75–200, 1974.

[31] C. Bény, "Deep learning and the renormalization group," arXiv:1301.3124, Jan. 2013.

[32] M. Koch-Janusz and Z. Ringel, "Mutual information, neural networks and the renormalization group," Nature Physics, vol. 14, no. 6, pp. 578–582, 2018.

[33] B. McCoy and T. T. Wu, The Two-Dimensional Ising Model. Harvard University Press, 1973.

[34] J. Carrasquilla and R. G. Melko, "Machine learning phases of matter," Nature Physics, vol. 13, no. 5, pp. 431–434, 2017.

[35] A. Morningstar and R. G. Melko, "Deep learning the ising model near criticality," The Journal of Machine Learning Research, vol. 18, no. 1, pp. 5975–5991, 2017.

[36] S. Saremi and T. J. Sejnowski, "Hierarchical model of natural images and the origin of scale invariance," Proceedings of the National Academy of Sciences of the United States of America, vol. 110, no. 8, pp. 3071–3076, Feb. 2013.

[37] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," Colorado Univ at Boulder Dept of Computer Science, Tech. Rep., 1986.

[38] Y. Freund and D. Haussler, "Unsupervised learning of distributions on binary vectors using two layer networks," in Advances in neural information processing systems, 1992, pp. 912–919.

[39] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," Neural computation, vol. 14, no. 8, pp. 1771–1800, 2002.

[40] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," Proceedings of the national academy of sciences, vol. 79, no. 8, pp. 2554–2558, 1982.

[41] G. E. Hinton, "A practical guide to training restricted boltzmann machines," in Neural networks: Tricks of the trade. Springer, 2012, pp. 599–619.

[42] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning." in Aistats, vol. 10. Citeseer, 2005, pp. 33–40.

[43] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in Proceedings of the 24th International Conference on Machine Learning. New York, NY, USA: ACM, 2007, pp. 791–798, event-place: Corvalis, Oregon, USA. [Online]. Available: http://doi.acm.org/10.1145/1273496.1273596

[44] M. Ranzato, A. Krizhevsky, and G. Hinton, "Factored 3-way restricted boltzmann machines for modeling natural images," in Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Mar. 2010, pp. 621–628. [Online]. Available: http://proceedings.mlr.press/v9/ranzato10a.html

[45] T. Tieleman, "Training restricted boltzmann machines using approximations to the likelihood gradient," in Proceedings of the 25th international conference on Machine learning, 2008, pp. 1064–1071.

[46] I. Sutskever and T. Tieleman, "On the convergence properties of contrastive divergence," in Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010, pp. 789–795.

[47] C. G. Callan Jr, "Broken scale invariance in scalar field theory," Physical Review D, vol. 2, no. 8, p. 1541, 1970.

[48] V. Loreto, L. Pietronero, A. Vespignani, and S. Zapperi, "Renormalization group approach to the critical behavior of the forest-fire model," Physical review letters, vol. 75, no. 3, p. 465, 1995.

[49] S. Clar, B. Drossel, and F. Schwabl, "Forest fires and other examples of self-organized criticality," Journal of Physics: Condensed Matter, vol. 8, no. 37, p. 6803, 1996.

[50] A. Diaz-Guilera, "Dynamic renormalization group approach to self-organized critical phenomena," EPL (Europhysics Letters), vol. 26, no. 3, p. 177, 1994.

[51] D. Poland, S. Rychkov, and A. Vichi, "The conformal bootstrap: Theory, numerical techniques, and applications," Reviews of Modern Physics, vol. 91, no. 1, p. 015002, 2019.

[52] R. Ruger, "Ssmc - monte carlo simulation code for classical spin systems like the ising model," https://github.com/robertrueger/SSMC, 2014.

[53] F. Chollet, D. Falbel, J. Allaire, Y. Tang, W. Van Der Bijl, M. Studer, RStudio, and Google, "Keras," https://github.com/fchollet/keras, 2015.

[54] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," nature, vol. 323, no. 6088, pp. 533–536, 1986.

[55] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.

[56] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," arXiv preprint arXiv:1901.06032, 2019.

[57] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in Artificial intelligence and statistics, 2016, pp. 464–472.

[58] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," in International conference on rough sets and knowledge technology. Cham: Springer, 2014, pp. 364–375.

[59] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2921–2929.

[60] G. Vidal, "Entanglement renormalization," Physical review letters, vol. 99, no. 22, p. 220405, 2007.

[61] G. Evenbly and G. Vidal, "Tensor network renormalization," Physical review letters, vol. 115, no. 18, p. 180405, 2015.

[62] A. Cichocki, A.-H. Phan, Q. Zhao, N. Lee, I. Oseledets, M. Sugiyama, D. P. Mandic et al., "Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives," Foundations and Trends® in Machine Learning, vol. 9, no. 6, pp. 431–673, 2017.

[63] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev et al., "Grandmaster level in starcraft ii using multi-agent reinforcement learning," Nature, pp. 1–5, 2019.

[64] L. Wang, "Discovering phase transitions with unsupervised learning," Phys. Rev. B, vol. 94, no. 19, p. 195105, 2016.

[65] D.-L. Deng, X. Li, and S. D. Sarma, "Machine learning topological states," Phys. Rev. B, vol. 96, no. 19, p. 195145, 2017.

[66] P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, "Machine learning quantum phases of matter beyond the fermion sign problem," Scientific reports, vol. 7, no. 1, p. 8823, 2017.

[67] K.-I. Aoki and T. Kobayashi, "Restricted boltzmann machines for the long range ising models," Modern Physics Letters B, vol. 30, no. 34, p. 1650401, 2016.

[68] Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, "Restricted boltzmann machine learning for solving strongly correlated quantum systems," Phys. Rev. B, vol. 96, no. 20, p. 205152, 2017.

[69] E. Howard, "Holographic renormalization with machine learning," arXiv:1803.11056, 2018.

[70] E. De Mello Koch, R. De Mello Koch, and L. Cheng, "Is deep learning a renormalization group flow?" IEEE Access, vol. 8, pp. 106 487–106 505, 2020.

[71] S.-H. Li and L. Wang, "Neural network renormalization group," Phys. Rev. Lett., vol. 121, no. 26, p. 260601, 2018.

[72] D. Kim and D.-H. Kim, "Smallest neural network to learn the ising criticality," Phys. Rev. E, vol. 98, no. 2, p. 022138, 2018.

[73] C. J. Geyer, "Practical markov chain monte carlo," Statistical science, pp. 473–483, 1992.

[74] K. G. Wilson and J. Kogut, "The renormalization group and the ϵ expansion," Physics reports, vol. 12, no. 2, pp. 75–199, 1974.

[75] E. Efrati, Z. Wang, A. Kolan, and L. P. Kadanoff, "Real-space renormalization in statistical mechanics," Reviews of Modern Physics, vol. 86, no. 2, p. 647, 2014.

[76] L. P. Kadanoff, A. Houghton, and M. C. Yalabik, "Variational approximations for renormalization group transformations," Journal of Statistical Physics, vol. 14, no. 2, pp. 171–203, 1976.

[77] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in Proceedings of the 26th annual international conference on machine learning, 2009, pp. 609–616.

[78] S. Hochreiter and J. Schmidhuber, "Flat minima," Neural Computation, vol. 9, no. 1, pp. 1–42, 1997.

[79] G. E. Hinton and D. Van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in Proceedings of the sixth annual conference on Computational learning theory, 1993, pp. 5–13.

[80] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," arXiv preprint arXiv:1609.04836, 2016.

[81] J. Langford and R. Caruana, "(not) bounding the true error," Advances in Neural Information Processing Systems, vol. 14, pp. 809–816, 2001.

[82] D. A. McAllester, "Some pac-bayesian theorems," Machine Learning, vol. 37, no. 3, pp. 355–363, 1999.

[83] B. Neyshabur, S. Bhojanapalli, and N. Srebro, "A pac-bayesian approach to spectrally-normalized margin bounds for neural networks," arXiv preprint arXiv:1707.09564, 2017.

[84] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, "Exploring generalization in deep learning," in Advances in neural information processing systems, 2017, pp. 5947–5956.

[85] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," Journal of Machine Learning Research, vol. 3, no. Nov, pp. 463–482, 2002.

[86] B. Neyshabur, R. Tomioka, and N. Srebro, "Norm-based capacity control in neural networks," in Conference on Learning Theory, 2015, pp. 1376–1401.

[87] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, "Spectrally-normalized margin bounds for neural networks," in Advances in Neural Information Processing Systems, 2017, pp. 6240–6249.

[88] N. Golowich, A. Rakhlin, and O. Shamir, "Size-independent sample complexity of neural networks," in Conference On Learning Theory. PMLR, 2018, pp. 297–299.

[89] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz, and M. Botvinick, "On the importance of single directions for generalization," arXiv preprint arXiv:1803.06959, 2018.

[90] K. G. Wilson, "The renormalization group: Critical phenomena and the kondo problem," Reviews of modern physics, vol. 47, no. 4, p. 773, 1975.

[91] S. R. White, "Strongly correlated electron systems and the density matrix renormalization group," Physics Reports, vol. 301, no. 1-3, pp. 187–204, 1998.

[92] A. Degenhard and J. Rodríguez-Laguna, "Real-space renormalization-group approach to field evolution equations," Physical Review E, vol. 65, no. 3, p. 036703, 2002.

[93] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," Neural networks, vol. 2, no. 1, pp. 53–58, 1989.

[94] E. Plaut, "From principal subspaces to principal components with linear autoencoders," arXiv preprint arXiv:1804.10253, 2018.

[95] L. Jing, J. Zbontar, and Y. LeCun, "Implicit rank-minimizing autoencoder," arXiv preprint arXiv:2010.00679, 2020.

[96] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[97] T. T. Team. (2019, jan) Flowers. [Online]. Available: http://download.tensorflow.org/example_images/flower_photos.tgz

[98] S. Dev, Y. H. Lee, and S. Winkler, "Color-based segmentation of sky/cloud images from ground-based cameras," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 10, no. 1, pp. 231–242, 2017.

[99] S. S. Funai and D. Giataganas, "Thermodynamics and feature extraction by machine learning," arXiv preprint arXiv:1810.08179, 2018.

[100] H. W. Lin, M. Tegmark, and D. Rolnick, "Why does deep and cheap learning work so well?" Journal of Statistical Physics, vol. 168, no. 6, pp. 1223–1247, 2017.

[101] L. P. Kadanoff, Statistical physics: statics, dynamics and renormalization. World Scientific Publishing Company, 2000.

Appendix A

Derivations

A.1 RBM expectation values

The expectation values quoted in equations (2.7), (2.8) and (2.9) are derived using (2.2). Data expectation values are evaluated by summing over all samples, $\hat{v}_i^{(A)}$ in the training set. On the other hand model expectation values employ sums over the entire space of visible and hidden vectors. This is such an enormous sum that its numerically intractable. Consequently, the approximations described in Section 2.2.1 are used. The complete set of expectation values needed to describe the RBM are given by

$$\langle v_i h_a \rangle_{data} = \frac{1}{N_s} \sum_{A=1}^{N_s} \hat{v}_i^{(A)} \tanh\left(\sum_k W_{ka} \hat{v}_k^{(A)} + b_a^{(h)} \right), \tag{A.1}$$

$$\langle v_i h_a \rangle_{model} = \sum_{\{\mathbf{v},\mathbf{h}\}} \tanh\left(\sum_k W_{ka} v_k + b_a^{(h)} \right) \cdot \tanh\left(\sum_j W_{ij} h_j + b_i^{(v)} \right), \tag{A.2}$$

$$\langle v_i \rangle_{data} = \frac{1}{N_s} \sum_{A=1}^{N_s} \hat{v}_i^{(A)}, \tag{A.3}$$

$$\langle v_i \rangle_{model} = \sum_{\{\mathbf{h}\}} \tanh\left(\sum_a W_{ia} h_a + b_i^{(v)} \right), \tag{A.4}$$

$$\langle h_a \rangle_{data} = \frac{1}{N_s} \sum_{A=1}^{N_s} \tanh\left(\sum_i W_{ia} \hat{v}_i^{(A)} + b_a^{(h)} \right), \tag{A.5}$$

$$\langle h_a \rangle_{model} = \sum_{\{\mathbf{v}\}} \tanh\left(\sum_i W_{ia} v_i + b_a^{(h)} \right), \tag{A.6}$$

with $\hat{v}_i^{(A)}$ the Ath sample of the data set, $\hat{\mathbf{v}}$.

A.2 Two Versions of RG

In this section we review two versions of the RG that are needed in this article. The first of these, the variational renormalization group, was introduced by Kadanoff [101, 76, 75] as a method to approximately perform the renormalization group in practice. The second version, block spin averaging is a discrete version of RG. This version of RG is useful in settings we study in this thesis as well as more general unsupervised deep learning applications.

A.2.1 Variational RG

Consider a system of N spins $\{v_i\}$ which each take the values ± 1. The partition function describing the system is given by

$$Z = \sum_{v_i} e^{-H(\{v_i\})}. \tag{A.7}$$

Here the sum is over all possible configurations of the system of spins and the function $H(\{v_i\})$, called the Hamiltonian, gives the energy of the system. This would include the energy of each individual spin as well as the energy associated to the fact that the collection of spins is interacting. The Hamiltonian $H(\{v_i\})$ can be an arbitrarily complicated function of the spins

$$\begin{aligned} H(\{v_i\}) = &- \sum_i K_i v_i - \sum_{i,j} K_{ij} v_i v_j \\ &- \sum_{i,j,k} K_{ijk} v_i v_j v_k + \cdots . \end{aligned} \tag{A.8}$$

The RG flow maps the original Hamiltonian to a new Hamiltonian with a different set of coupling constants. The new Hamiltonian

$$H(\{h_a\}) = -\sum_a K'_a h_a - \sum_{a,b} K'_{ab} h_a h_b - \sum_{a,b,c} K'_{abc} h_a h_b h_c + \cdots, \tag{A.9}$$

gives the energy for the coarse grained spins h_a. After many RG iterations many coupling constants (the so called irrelevant terms) flow to zero. A much smaller number may remain constant (marginal terms) or even grow (relevant terms). To implement this conceptual framework a concrete RG mapping is needed. Variational RG provides a mapping which is not exact but can be implemented numerically. It does this by introducing an operator $T_\lambda(\{v_i, h_a\})$ which is a function of a set of parameters $\{\lambda\}$. The Hamiltonian after a step of RG flow is

$$e^{-H_{RG}(\{h_a\})} = \sum_{v_i} e^{T_\lambda(\{v_i,h_a\}) - H(\{v_i\})}. \tag{A.10}$$

The form of $T_\lambda(\{v_i, h_a\})$ must be chosen cleverly, for each problem we consider. This is the tough step in variational RG and it is carried out using physical intuition, but essentially on a trial and error basis. Once a given $T_\lambda(\{v_i, h_a\})$ has been chosen, we minimize the following quantity by choosing the parameters $\{\lambda\}$

$$\log(\sum_{v_i} e^{-H(\{v_i\})}) - \log(\sum_{h_a} e^{-H_{RG}(\{h_a\})}). \tag{A.11}$$

The minimum possible value for this quantity is zero. Notice that when

$$\sum_{h_a} e^{T_\lambda(\{v_i,h_a\})} = 1, \tag{A.12}$$

(A.11) attains its minimum value of 0 and the RG transformation is called exact.

A.2.2 Block spin averaging

Block spin averaging is a pedagogical version of RG. To illustrate the method, consider a rectangular lattice of interacting spins. Divide the lattice into blocks of 2×2 squares. Block spin averaging describes the system in terms of *block variables*, which are variables describing the average behavior of each block. The "block spin" is literally the average of

the four spins in the block. The plots shown in Figures 2.7a use block spin averaging. The block spins h_a are each an average of four visible spins v_i.

A.3 Equivalence of maximum log likelihood estimation and minimization of Kullback-Liebler divergence

Maximum log likelihood estimation (MLE) is a widely used method to determine model parameters. Using MLE, model parameters are determined by maximizing the log likelihood that the model $p(\mathbf{v}|\theta)$, with parameters θ, produced the given data $\mathbf{v}$

$$\theta_{MLE} = \underset{\theta}{\text{argmax}}\ \log(p(\mathbf{v}|\theta)). \tag{A.13}$$

The Kullback Liebler (KL) divergence determines the difference between two distributions, which is used to match the data distribution $q(\mathbf{v})$ to the model distribution $p(\mathbf{v}|\theta)$. Model parameters are determined by minimizing the KL divergence between the model and data distributions

$$\begin{aligned}\theta_{minKL} =& \underset{\theta}{\text{argmin}}\ E_q[\log(q(\mathbf{v})) - \log(p(\mathbf{v}|\theta))] \\ =& \underset{\theta}{\text{argmin}}\ (E_q[\log(q(\mathbf{v}))] - E_q[\log(p(\mathbf{v}|\theta))]).\end{aligned} \tag{A.14}$$

The first term $E_q[\log(q(\mathbf{v}))]$ does not have any θ dependance so removing this term gives

$$\theta_{minKL} = \underset{\theta}{\text{argmin}}\ -E_q[\log(p(\mathbf{v}|\theta))], \tag{A.15}$$

which we can change to an argmax

$$\theta_{minKL} = \underset{\theta}{\text{argmax}}\ E_q[\log(p(\mathbf{v}|\theta))]. \tag{A.16}$$

We know that averaging $\log(p(v_i|\theta))$ over enough samples, v_i, where $i = 1, 2, \ldots, n$ as $n \to \infty$, will give us the expected value of $\log(p(v_i|\theta))$ using the law of large numbers. Using this,

equation (A.13) becomes

$$\begin{aligned}\theta_{MLE} =& \underset{\theta}{\text{argmax}} \lim_{n\to\infty} \sum_{i=1}^{n} \log(p(v_i|\theta)) \\ =& \underset{\theta}{\text{argmax}}\, E_q[\log(p(\mathbf{v}|\theta)] \\ =& \theta_{minKL}.\end{aligned} \tag{A.17}$$

A.4 RBM learning rule

The goal of training an RBM is to match the RBM model probability distribution to that of the data. If the model has learnt the probability distribution of the data, the updates will be zero. The parameter updates performed during training consist of model averages and data averages. By deriving the learning rule from the KL divergence we will see how these averages appear. We know the distribution of the data $q(\mathbf{v})$. The unknowns in the model distribution, $p(\mathbf{v})$, are the parameters $\mathbf{b}^{(\mathbf{v})}$, $\mathbf{b}^{(\mathbf{h})}$ and $\mathbf{W}$

$$p(\mathbf{v}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}, \tag{A.18}$$

with

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}. \tag{A.19}$$

We can write the update rule analytically using equations (3.12) and (A.18) as

$$\frac{\partial D_{KL}}{\partial W_{ia}} = -\sum_{\mathbf{v}} \frac{q(\mathbf{v})}{p(\mathbf{v})} \frac{\partial p(\mathbf{v})}{\partial W_{ia}} = \Delta_{W_{ia}}, \tag{A.20}$$

$$\frac{\partial D_{KL}}{\partial b_i^{(v)}} = -\sum_{\mathbf{v}} \frac{q(\mathbf{v})}{p(\mathbf{v})} \frac{\partial p(\mathbf{v})}{\partial b_i^{(v)}} = \Delta_{b_i^{(v)}}, \tag{A.21}$$

$$\frac{\partial D_{KL}}{\partial b_a^{(h)}} = -\sum_{\mathbf{v}} \frac{q(\mathbf{v})}{p(\mathbf{v})} \frac{\partial p(\mathbf{v})}{\partial b_a^{(h)}} = \Delta_{b_a^{(h)}}. \tag{A.22}$$

We can arrive at update rules which relate averages for the model and averages for the data by calculating derivatives of $p(\mathbf{v})$ with respect to the model parameters W_{ia}, $b_i^{(v)}$ and $b_a^{(h)}$ as

follows

$$\begin{aligned}
\frac{\partial p(\mathbf{v})}{\partial W_{ia}} =& \frac{\partial}{\partial W_{ia}}\left[\frac{1}{Z}\sum_{\mathbf{h}} e^{-E}\right] \\
=& \frac{\partial}{\partial W_{ia}}\left[\frac{\sum_{\mathbf{h}} e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}}\right] \\
=& -\frac{\sum_{\mathbf{h}} v_i h_a e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} + \\
& + \frac{\sum_{\mathbf{h}} e^{-E}}{\left(\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}\right)^2}\sum_{\mathbf{h}}\sum_{\mathbf{v}} h_a v_i e^{-E} \\
=& -\frac{v_i h_a \sum_{\mathbf{h}} e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} + p(\mathbf{v})\frac{\sum_{\mathbf{h}}\sum_{\mathbf{v}} v_i h_a e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} \\
=& -\frac{v_i h_a \sum_{\mathbf{h}} e^{-E}}{Z} + p(\mathbf{v})\frac{\sum_{\mathbf{h}}\sum_{\mathbf{v}} v_i h_a e^{-E}}{Z} \\
=& -v_i h_a p(\mathbf{v}) + p(\mathbf{v})\langle v_i h_a\rangle_p,
\end{aligned} \tag{A.23}$$

$$\begin{aligned}
\frac{\partial p(\mathbf{v})}{\partial b_i^{(v)}} =& \frac{\partial}{\partial b_i^{(v)}}\left[\frac{1}{Z}\sum_{\mathbf{h}} e^{-E}\right] \\
=& \frac{\partial}{\partial b_i^{(v)}}\left[\frac{\sum_{\mathbf{h}} e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}}\right] \\
=& -\frac{\sum_{\mathbf{h}} v_i e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} + \\
& + \frac{\sum_{\mathbf{h}} e^{-E}}{\left(\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}\right)^2}\sum_{\mathbf{h}}\sum_{\mathbf{v}} v_i e^{-E} \\
=& -\frac{v_i \sum_{\mathbf{h}} e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} + p(\mathbf{v})\frac{\sum_{\mathbf{h}}\sum_{\mathbf{v}} v_i e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} \\
=& -\frac{v_i \sum_{\mathbf{h}} e^{-E}}{Z} + p(\mathbf{v})\frac{\sum_{\mathbf{h}}\sum_{\mathbf{v}} v_i e^{-E}}{Z} \\
=& -v_i p(\mathbf{v}) + p(\mathbf{v})\langle v_i\rangle_p,
\end{aligned} \tag{A.24}$$

$$\begin{aligned}
\frac{\partial p(\mathbf{v})}{\partial b_a^{(h)}} &= \frac{\partial}{\partial b_a^{(h)}}\left[\frac{1}{Z}\sum_{\mathbf{h}} e^{-E}\right] \\
&= \frac{\partial}{\partial b_a^{(h)}}\left[\frac{\sum_{\mathbf{h}} e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}}\right] \\
&= -\frac{\sum_{\mathbf{h}} h_a e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} + \\
&\quad + \frac{\sum_{\mathbf{h}} e^{-E}}{(\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E})^2}\sum_{\mathbf{h}}\sum_{\mathbf{v}} h_a e^{-E} \\
&= -\frac{h_a\sum_{\mathbf{h}} e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} + p(\mathbf{v})\frac{\sum_{\mathbf{h}}\sum_{\mathbf{v}} h_a e^{-E}}{\sum_{\mathbf{h}}\sum_{\mathbf{v}} e^{-E}} \\
&= -\frac{h_a\sum_{\mathbf{h}} e^{-E}}{Z} + p(\mathbf{v})\frac{\sum_{\mathbf{h}}\sum_{\mathbf{v}} h_a e^{-E}}{Z} \\
&= -h_a p(\mathbf{v}) + p(\mathbf{v})\langle h_a\rangle_p.
\end{aligned} \tag{A.25}$$

Averages related to the model are denoted by $\langle\cdot\rangle_p$ and averages related to the data by $\langle\cdot\rangle_q$. Substituting equations (A.23), (A.24) and (A.25) into equations (A.20), (A.21) and (A.22) we get

$$\begin{aligned}
\frac{D_{KL}}{\partial W_{ia}} &= -\sum_{\mathbf{v}}\frac{q(\mathbf{v})}{p(\mathbf{v})}(-v_i h_a p(\mathbf{v}) + p(\mathbf{v})\langle v_i h_a\rangle_p) \\
&= -\sum_{\mathbf{v}}(-q(\mathbf{v})v_i h_a + q(\mathbf{v})\langle v_i h_a\rangle_p) \\
&= \langle v_i h_a\rangle_q - \langle v_i h_a\rangle_p,
\end{aligned} \tag{A.26}$$

$$\begin{aligned}
\frac{D_{KL}}{\partial b_i^{(v)}} &= -\sum_{\mathbf{v}}\frac{q(\mathbf{v})}{p(\mathbf{v})}(-v_i p(\mathbf{v}) + p(\mathbf{v})\langle v_i\rangle_p) \\
&= -\sum_{\mathbf{v}}(-q(\mathbf{v})v_i + q(\mathbf{v})\langle v_i\rangle_p) \\
&= \langle v_i\rangle_q - \langle v_i\rangle_p,
\end{aligned} \tag{A.27}$$

$$\begin{aligned}
\frac{D_{KL}}{\partial b_a^{(h)}} &= -\sum_{\mathbf{v}}\frac{q(\mathbf{v})}{p(\mathbf{v})}(-h_a p(\mathbf{v}) + p(\mathbf{v})\langle h_a\rangle_p) \\
&= -\sum_{\mathbf{v}}(-q(\mathbf{v})h_a + q(\mathbf{v})\langle h_a\rangle_p) \\
&= \langle h_a\rangle_q - \langle h_a\rangle_p.
\end{aligned} \tag{A.28}$$

From this derivation we can see that the KL divergence compares expectation values of the model to expectation values of the data to determine the most suitable parameters for the RBM. Equations (A.20), (A.21) and (A.22) will equal zero when training halts i.e. the network's parameter updates ΔW_{ia}, $\Delta b_i^{(v)}$ and $\Delta b_a^{(h)}$ are zero.

Appendix B

Supporting technical details

B.1 Restricted Boltzmann machines

Restricted Boltzmann Machines (RBMs) consist of two layers of nodes - a visible layer (input nodes) and a hidden layer (output nodes). Connections are allowed between every visible and hidden node but no connections are allowed between nodes within the same layer. The energy function for the RBM is given by

$$E = -\sum_{i=1}^{N_v}\sum_{a=1}^{N_h} v_i W_{ia} h_a - \sum_{i=1}^{N_v} v_i b_i^{(v)} - \sum_{a=1}^{N_h} h_a b_a^{(h)}, \tag{B.1}$$

where W_{ia} is the weight matrix which determines strengths between the ith visible, v_i, and ath hidden node, h_a, $b_i^{(v)}$ is the bias of the ith visible node and $b_a^{(h)}$ the bias of the ath hidden node. The visible and hidden nodes take values of ± 1.

The probability of a given visible and hidden vector is

$$p(v,h) = \frac{1}{Z} e^{-E}, \tag{B.2}$$

where the normalization Z is the partition function

$$Z = \sum_{\{v,h\}} e^{-E}. \tag{B.3}$$

The marginal distribution of a visible or hidden vector is given by summing over the space of hidden and visible vectors respectively

$$p(v) = \frac{1}{Z} \sum_{\{h\}} e^{-E}, \tag{B.4}$$

$$p(h) = \frac{1}{Z}\sum_{\{v\}} e^{-E}. \tag{B.5}$$

The network is trained using a technique called contrastive divergence which approximates minimizing the Kullback-Liebler divergence. The goal of training is to match the distribution of the model to the distribution of the data. The Kullback-Libeler divergence measures how similar two distributions are and is given by

$$\begin{aligned} D_{KL}(q||p) &= \sum_{\{v\}} q(v)(\log(q(v)) - log(p(v))) \\ &= \sum_{\{v\}} q(v)\log\left(\frac{q(v)}{p(v)}\right), \end{aligned} \tag{B.6}$$

where $q(v)$ defines the data distribution and $p(v)$ denotes the model distribution we determine during training. The Kullback-Libeler divergence is always not negative and only vanishes if the two distributions are equal. Training tunes the model parameters W_{ia}, $b_i^{(v)}$ and $b_a^{(h)}$ to minimize D_{KL}. We tune parameters by taking derivatives of D_{KL} with respect to each parameter

$$\frac{\partial D_{KL}}{\partial W_{ia}} = \langle v_i h_a \rangle_{data} - \langle v_i h_a \rangle_{model}, \tag{B.7}$$

$$\frac{\partial D_{KL}}{\partial b_i^{(v)}} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model}, \tag{B.8}$$

$$\frac{\partial D_{KL}}{\partial b_a^{(h)}} = \langle h_a \rangle_{data} - \langle h_a \rangle_{model}. \tag{B.9}$$

The right hand side of these expressions can now be evaluated. It is impractical to perform the sum over the state space of visible and hidden vectors, because the size of each state space is enormous. It is for this reason that we employ contrastive divergence (CD) to approximate the KL divergence. CD allows us to sum over the space of input vectors present in the training set rather than the entire state space of visible vectors. To approximate the state space of hidden vectors we sample a set of hidden vectors, generated using the input (visible) vectors.

Given a set of visible vectors we can sample the hidden vectors using the probability

$$p(h_a = 1|v) = \frac{1}{2}\left(1 + \tanh\left(\sum_i W_{ia} v_i + b_a^{(h)}\right)\right). \tag{B.10}$$

Given a set of hidden vectors we can sample a set of visible vectors using the probability

$$p(v_i = 1|h) = \frac{1}{2}\left(1 + \tanh\left(\sum_a W_{ia} h_a + b_i^{(v)}\right)\right). \tag{B.11}$$

To calculate expectation values over the data we use the input training set of visible vectors denoted $\hat{v}$. To generate the set of hidden vectors used in the data averages, $\hat{h}$, we sample hidden vectors using equation (B.10) and $\hat{v}$.

We then sample the set of visible vectors which are used in the model averages, $\tilde{v}$, using $\hat{h}$ and equation (B.11). Using $\tilde{v}$ and equation (B.10) we sample hidden nodes to be used in the model expectations, $\tilde{h}$.

Contrastive divergence now amounts to the following sequence of approximations

$$\langle v_i h_a \rangle_{data} = \frac{1}{N_s} \sum_{A=1}^{N_s} \hat{v}_i^{(A)} \hat{h}_a^{(A)}, \tag{B.12}$$

$$\langle v_i h_a \rangle_{model} = \frac{1}{N_s} \sum_{A=1}^{N_s} \tilde{v}_i^{(A)} \tilde{h}_a^{(A)}, \tag{B.13}$$

$$\langle v_i \rangle_{data} = \frac{1}{N_s} \sum_{A=1}^{N_s} \hat{v}_i^{(A)}, \tag{B.14}$$

$$\langle v_i \rangle_{model} = \frac{1}{N_s} \sum_{A=1}^{N_s} \tilde{v}_i^{(A)}, \tag{B.15}$$

$$\langle h_a \rangle_{data} = \frac{1}{N_s} \sum_{A=1}^{N_s} \hat{h}_a^{(A)}, \tag{B.16}$$

$$\langle h_a \rangle_{model} = \frac{1}{N_s} \sum_{A=1}^{N_s} \tilde{h}_a^{(A)}. \tag{B.17}$$

B.2 RG

In this Appendix we will give a brief review of the renormalization group. The goal of the renormalization group is to derive a long distance effective description, given some underlying microscopic dynamics. This typically happens by coarse graining the short distance degrees of freedom. The name "the renormalization group" is a bad one since the mathematical structure of the renormalization group is not that of a group, renormalization of quantum field theory is not an essential ingredient[1] and the method is not unique but

[1] One of the earliest applications of these methods is to quantum field theory, which is the origin of the name.

rather it is a set of ideas that must adapted to the nature of the problem at hand. In the next section we will review the method as it applies to Euclidean quantum field theory. We have chosen to review this material because it is the basic mechanism at work in fully connected layers and, further, it is a good example of how the renormalization group explains why so many parameters of the short distance theory simply have no effect on the effective description. We have interpreted this as the basic mechanism behind understanding why deep networks generalize. We go on to describe the block spin transformation implementation of the renormalization group coarse graining as it is directly relevant to the discussion of the Ising model in Section 4.1.1.

B.2.1 Momentum space RG

Momentum space RG is a tool used routinely in quantum field theory and statistical mechanics [74]. The method coarse grains by first organizing the theory according to length scales and then averaging over the short distance degrees of freedom, which gives an effective theory for the long distance degrees of freedom.

To illustrate the procedure, we will consider a quantum field theory, describing a quantum field ϕ. Observables $\mathcal{O}$ are functions (usually polynomials) of the field and its derivatives. Examples of observables are the energy or momentum of the field, or correlation functions of the field. To calculate the expected value $\langle \mathcal{O} \rangle$ of observable $\mathcal{O}$, integrate (i.e. average) over all possible field configurations

$$\langle \mathcal{O} \rangle = \int [d\phi] e^{-S} \mathcal{O}. \tag{B.18}$$

The factor e^{-S}, which defines a probability measure on the space of fields, depends on the theory considered. S is called the action of the theory. The action is a polynomial in the field and its derivatives. The coefficients appearing in this polynomial are the parameters of the theory, things like couplings and masses. These are the parameters of the short distance theory. A theory is defined by specifying the action S.

To coarse grain, express the position space field in terms of momentum space components

$$\phi(x) = \int dk e^{ik \cdot x} \phi(k), \tag{B.19}$$

$e^{ik \cdot x}$ oscillates in position space with wavelength $\frac{2\pi}{k}$. High momentum (big k) components have small wavelengths and encode small distance structure. Low momentum components have huge wavelengths and describe large distance structure. Declare there is a smallest possible structure, implemented by cutting off the momentum modes at a large momentum Λ

as follows

$$[d\phi] = \prod_{k<\Lambda} d\phi(k). \tag{B.20}$$

The renormalization group breaks the integration measure into high and low momentum components $[d\phi] = [d\phi_<][d\phi_>]$ where

$$\begin{aligned} [d\phi_<] &= \prod_{k<(1-\varepsilon)\Lambda} d\phi(k) \\ [d\phi_>] &= \prod_{(1-\varepsilon)\Lambda<k<\Lambda} d\phi(k), \end{aligned} \tag{B.21}$$

By assumption, we consider observables that depend only on large scale structure of the theory, i.e. observables that depend only on $\phi_<$. In this case, when computing the expected value of $\mathscr{O}$ we can pull $\mathscr{O}$ out of the integral over $\phi_>$ and integrate over the high momentum components

$$\begin{aligned} \langle \mathscr{O}(\phi_<) \rangle &= \int [d\phi_<] \int [d\phi_>]\, e^{-S} \mathscr{O}(\phi_<) \\ &= \int [d\phi_<]\, e^{-S_{\text{eff}}} \mathscr{O}(\phi_<), \end{aligned} \tag{B.22}$$

This procedure of splitting momentum components into two sets and integrating over the large momenta defines a new action S_{eff}. Repeating the procedure many times defines the RG flow under which S_{eff} changes continuously. After the flow, one is left with an integral over the very long wavelength modes. This completes the coarse graining: we have a new theory defined by S_{eff}. The new theory uses only long wavelength components of the field and correctly reproduces the expected value of any observable depending only on long wavelength components.

Values of the parameters of the theory, which appear in S_{eff} change under this transformation. In general, many possible terms are generated and appear in S_{eff}. Each possible term defines a coupling of the theory. Each coupling can be classified as marginal (the size of the coupling is unchanged by the RG flow), relevant (the coupling grows under the flow) or irrelevant (the coupling goes to zero under the flow). It is a dramatic insight of Wilson that almost all couplings in any given quantum field theory are irrelevant and so the low energy theory is characterized by a handful of parameters.

To compare how parameters have changed, we want to compare the action before and after integrating out a shell of modes. For this comparison to be meaningful, the integration domains, before and after integrating the shell out, must match. To achieve that we rescale

momenta (k), positions (x) and the field (ϕ) as follows

$$\begin{aligned} \vec{k} \to \vec{k}' &= \frac{\vec{k}}{1-\varepsilon} \\ \vec{x} \to \vec{x}' &= (1-\varepsilon)\vec{x} \\ \vec{\phi} \to \vec{\phi}' &= \frac{\vec{\phi}}{1-\varepsilon}, \end{aligned} \tag{B.23}$$

The rule for scaling momenta is fixed by matching integration domains. Since momentum is an inverse length, positions scale oppositely to momenta. The rule for how the field is scaled ensure that the free field theory, described by the action

$$S = \int d^4x \frac{1}{2} \vec{\nabla}\phi \cdot \vec{\nabla}\phi, \tag{B.24}$$

is invariant under the renormalization group flow. Here we have committed to $d = 4$ dimensions. The scaling of the field depends on d.

So there are two reasons why the couplings change: (i) we integrate over high momentum modes in (B.22)and (ii) we rescale according to (B.23). If we are very close to the free field fixed point couplings are weak and we can neglect the effect coming from doing the integral in (B.22). In this case we simply discard the high momentum modes and use the scaling (B.23). With the above transformation, a general term will scale as follows

$$\lambda \phi^n (\vec{\nabla}\phi \cdot \vec{\nabla}\phi)^m \to (1-\varepsilon)^{n+4m-4} \lambda \phi'^n (\vec{\nabla}'\phi' \cdot \vec{\nabla}'\phi')^m, \tag{B.25}$$

so that the coupling changes as

$$\lambda \to \lambda' = (1-\varepsilon)^{n+4m-4} \lambda. \tag{B.26}$$

Since $1-\varepsilon < 1$, the only relevant terms in the Lagrangian are

$$\phi, \qquad \phi^2, \qquad \phi^3, \tag{B.27}$$

The only marginal terms are[2]

$$\phi^4, \qquad \vec{\nabla}\phi \cdot \vec{\nabla}\phi. \tag{B.28}$$

[2]Quantum corrections, that is the corrections from actually integrating the shell of high momentum modes, make this term irrelevant. We call it marginally irrelevant.

There are an infinite number of irrelevant terms

$$\phi^{n+4}, \qquad \phi^n(\vec{\nabla}\phi\cdot\vec{\nabla}\phi)^m. \qquad m,n\geq 1 \tag{B.29}$$

Thus, although there are an infinite number of possible couplings in the microscopic short distance description, only five couplings in total survive in the long distance theory!
Key Idea: Near the free field fixed point, the renormalization group transformation simply discards the high momentum modes of the field that is being coarse grained. This is precisely what the trained deep network does.

B.2.2 Block spin transformations

The implementation of the renormalization group for the RBM trained on Ising data is block spin transformation as discussed in Section 4.1.1. Large groups of spins are averaged to produce the block spin. Any fluctuations of the spins on the lattice, that average to zero and are contained within the block being averaged, will be removed by the coarse graining. This implies that irrelevant perturbations are the ones that produce localized changes to the state, with the changes averaging to zero. This is in line with our intuition: irrelevant perturbations are associated with short length scales.

B.3 RGM

The connection between unsupervised deep learning and the renormalization group that we are arguing for in this study suggests economic parameterizations of the weight matrix. Understanding these parametrizations will shed light on the generalization puzzle. Since the specific form adopted for the weights is motivated by the renormalization group, we refer to the resulting deep network as a renormalization group machine, or RGM for short.

The economy of the parametrization follows because we know that only the big singular values count and that these only have low Fourier modes. We are going to build this into the weight matrix.

Using the singular value decomposition, we can write the weight matrix $W_{(m,a)}$ as

$$W_{ma}=\sum_{I=1}^{N_h} S_I|v,I\rangle_m\langle h,I|_a, \tag{B.30}$$

where $m=1,...,N_v=L_v^2$ and $a=1,...,N_h=L_h^2$. We have in mind input data given by two dimensional Ising lattices that have N_v sites arranged in an $L_v\times L_v$ square, or images that

have a total of N_v pixels. The data output by the network are states of a two dimensional Ising lattice with N_h sites, or images with a total of N_h pixels. We will use the language "visible lattice" and "hidden lattice" for both Ising and images in what follows. The index m runs over the sites of the visible lattice, so we can trade it for the coordinates (l,n) of a site in the visible lattice. Here l and n both run from 1 to L_v. Similarly, the index a runs over the sites of the hidden lattice, so we can trade it for the coordinates (b,c) of a site in the hidden lattice. Here b and c both run from 1 to L_h. With the new notation we can write the weight matrix W_{ma} as

$$W_{ma} \to W_{(l,n)\,(b,c)} = \sum_{I=1}^{N_h} S_I |v,I\rangle_{(l,n)} \langle h,I|_{(b,c)}.$$

Using the known simplifications we have

$$W_{(l,n)\,(b,c)} = \sum_{I=1}^{\kappa} S_I |v,I\rangle_{(l,n)} \langle h,I|_{(b,c)},$$

where κ is the number of large singular values and typically $\kappa \ll N_h$. We can also simplify the singular vectors appearing in the above expansion. To do this we use the discrete Fourier transform to write

$$|v,I\rangle_{(k,p)} = \sum_{m=1}^{L_v} \sum_{n=1}^{L_v} |v,I\rangle_{(m,n)} e^{-i2\pi(\frac{lk}{L_v}+\frac{np}{L_v})}, \tag{B.31}$$

for the visible vector and

$$|h,I\rangle_{(x,y)} = \sum_{a=1}^{L_h} \sum_{b=1}^{L_h} \langle h,I|_{(a,b)} e^{-i2\pi(\frac{ax}{L_h}+\frac{by}{L_h})}, \tag{B.32}$$

for the hidden vector. The inverse Fourier transform is

$$|v,I\rangle_{(l,n)} = \frac{1}{{L_v}^2} \sum_{k=1}^{L_v} \sum_{p=1}^{L_v} |v,I\rangle_{(k,p)}, e^{i2\pi(\frac{lk}{L_v}+\frac{np}{L_v})} \tag{B.33}$$

$$|h,I\rangle_{(a,b)} = \frac{1}{{L_h}^2} \sum_{x=1}^{L_h} \sum_{y=1}^{L_h} \langle h,I|_{(x,y)} e^{i2\pi(\frac{ax}{L_h}+\frac{by}{L_h})}. \tag{B.34}$$

The coarse graining interpretation implies that the singular vectors only have support at low Fourier modes. Consequently a more efficient parametrization of the visible singular vector is

$$|v,I\rangle_{(l,n)} = \frac{1}{{L_v}^2} \sum_{k=1}^{\alpha} \sum_{p=1}^{\alpha} C^I_{(k,p)} e^{i2\pi(\frac{lk}{L_v}+\frac{np}{L_v})}, \tag{B.35}$$

to form the visible vector $|v\rangle$. Typically $\alpha \ll L_h < L_v$. For the hidden vector, $\langle h|$, we use

$$\langle h, I|_{(a,b)} = \frac{1}{{L_h}^2} \sum_{x=1}^{\alpha} \sum_{y=1}^{\alpha} \frac{C^I_{(x,y)}}{2} . e^{-i2\pi(\frac{ax}{L_h}+\frac{by}{L_h})} \tag{B.36}$$

The fact that the same Fourier coefficients appear in the expansion of both the visible and the hidden singular vectors is a consequence of the fact that the network is performing a renormalization group transformation, so that the low frequency modes of the two vectors agree. Further, the fact that the singular vectors are real implies that

$$C^I_{(k,p)} = C^I_{(-k,-p)}. \tag{B.37}$$

Since the singular vector S_I appears as an overall factor, it can be absorbed into the definition of $C^I_{(k,p)}$. We will not do this, but it is worth noting since this observation implies that the S^I should not be counted when the number of effective parameters is counted. Thus, we finally obtain

$$W_{(m,n)\,(a,b)} = \frac{1}{{L_v}^2} \frac{1}{{L_h}^2} \sum_{I=1}^{\kappa} \sum_{k,p=1}^{\alpha} \sum_{x,y=1}^{\alpha} S_I \frac{C^I_{(k,p)} C^I_{(x,y)}}{2} e^{i2\pi\left((\frac{mk}{L_v}+\frac{np}{L_v})-(\frac{ax}{L_h}+\frac{by}{L_h})\right)}. \tag{B.38}$$

The only unknowns in the above formula for the weight matrix are the coefficients $C^I_{(k,p)}$ and the singular values S^I. To obtain these coefficients we compute the data covariance matrix for the training data set. The eigenvectors associated with the largest eigenvalues are identified with the visible singular vectors $|v, I\rangle_{(l,n)}$. By taking a Fourier transform of this vector and setting small Fourier modes to zero, we obtain the $C^I_{(k,p)}$.

Although we do not have a perfect formula for the singular values S_I, we have developed a simple rule that works reasonably well in practice. While the data determines the singular vectors, the singular values are determined by the coarse graining rule. Large singular values correspond to relevant modes and small singular values to irrelevant modes. To estimate the size of the singular value we estimate the magnitude of the vector

$$|I\rangle_{(r,s)} = \sum_{l,n=1}^{L_v} C_{(r,s)\,(l,n)} |v, I\rangle_{(l,n)}, \tag{B.39}$$

where $C_{(r,s)\,(l,n)}$ is a block spin transformation and $|v, I\rangle_{(l,n)}$ is a normalized singular vector.

The hidden and visible biases are chosen to be equal to the hidden and visible singular vectors associated to the largest singular vector. This choice is intuitively appealing: they bias configurations towards the subspace on which the data is concentrated. In addition, the trained biases are in remarkably good agreement with the singular vectors associated with the largest singular vectors.

The weakest part of our discussion has been in fixing the overall scales of the biases and the term involving the weights. Thanks to the activation functions in the network, much of the sensitivity to these scales is removed.

B.4 Ising model

The Ising model Hamiltonian is

$$H_I = -J \sum_{\langle \mathbf{i},\mathbf{j} \rangle} s_{\mathbf{i}} s_{\mathbf{j}}, \tag{B.40}$$

where the sum runs over nearest neighbors. In this study we take the spins to sit at the sites of a rectangular lattice. The probability distribution implied by the Hamiltonian is $e^{-\beta H}$ so that β and J are not independent parameters. We set $J = 1$ and use the temperature T to set $\beta = \frac{1}{T}$. By employing Monte Carlo methods we generate states of this 2D Ising lattice model. The training data set used to train the networks described in Section 4.1.1 is comprised of 40 000 samples of this lattice model. Each training epoch used the complete data set, and training was completed is roughly 6000 epochs.

B.5 Overlapping coarse graining

We can define many different coarse graining rules that use block spin transformations. Many of these rules result in overlap between different averaging blocks. The coarse graining rules at work in the trained RBM network is a quantitative match for block spin transformations with overlapping blocks, so it is important to understand the effect that overlap has if we are to understand what is happening in the network.

B.5.1 Generating the weight matrix

We create an effective weight matrix that, when applied to an input vector, will apply the coarse graining rule. The input vector is a 6400 dimensional vector, obtained by rearranging the elements of an 80×80 matrix. Each column of the weight matrix is constructed to average a single block of spins on the lattice. Thus, each column only has entries of 1's and

0's. For all coarse graining rules we move by two elements in the x and y direction for each new averaging block. This implies that the averaged vector is a 1600 dimensional vector.

The first block averaged is centered at the topmost and leftmost corner of the lattice. Any elements that should be averaged in the block, but do not exist in the input matrix, are discarded. This occurs for blocks averaging the edges of the input matrix.

B.5.2 Effect of overlapping

Weight matrices generated with block spins that have no overlap, have no variation in their singular values. For example, the weight matrix generated using an averaging block of size 2x2 has all singular values equal to 2. Each element in the output vector is obtained by averaging 4 elements from the input vector and each element in the input vector contributes to a unique element in the output vector. The hidden and visible singular vectors of the weight matrix are identical.

As overlap begins to occur, we see some variation in the singular values. The weight matrix used to plot figure B.1 has a coarse grain rule that uses averaging blocks with size 4×4, so that 16 spins are averaged to produce a single element in the output vector. In this case, the singular values decay almost linearly, as in Figure B.1. Since the block is moved by two elements for each new average, a single spin contributes to multiple outputs.

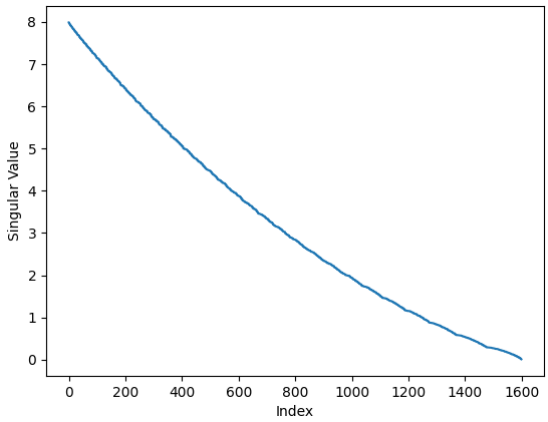


Fig. B.1 Singular values from a weight matrix generated from a coarse graining rule with little overlap.

Singular vectors corresponding to large singular values, shown in Figure B.2, are concentrated at low frequencies. Additionally, the visible and hidden singular vectors have the same shape. The hidden singular vectors agree perfectly with the visible singular vectors after rescaling by a factor of 2. For small singular values, shown in Figure B.3, the hidden

and visible singular vectors no longer agree and they are no longer concentrated at low frequencies.

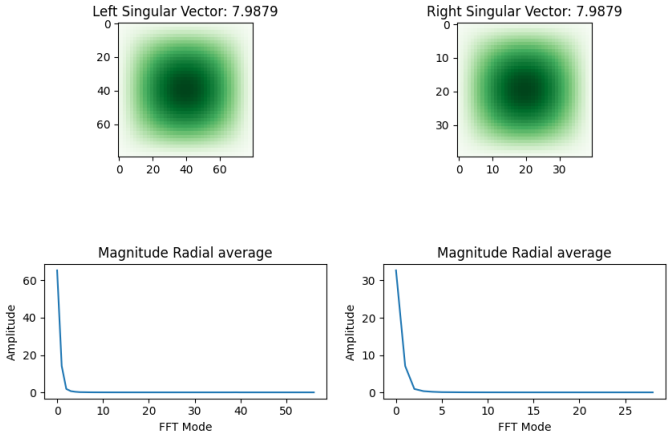


Fig. B.2 Singular vectors from a weight matrix generated from a coarse graining rule with little overlap for large singular values.

As the overlap between the averaging blocks grows, we see a small number of very large singular values that fall off rapidly. Figure B.4 is from a weight matrix with a block size of 20x20, averaging 400 elements for each output. The large block size means each element will contribute to a large number of output elements and so has a significant overlap between averaging blocks.

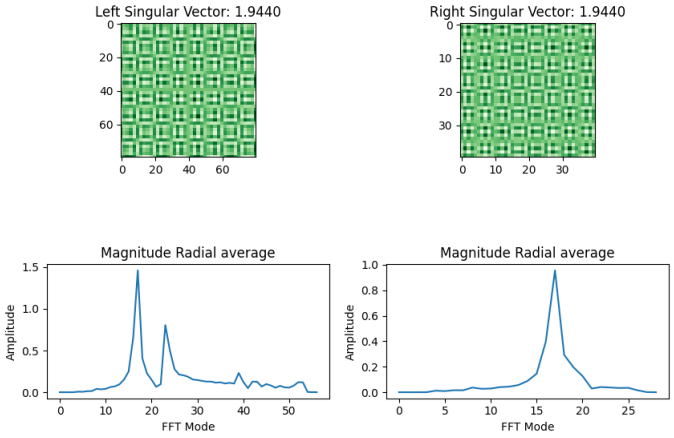


Fig. B.3 Singular vectors from a weight matrix generated from a coarse graining rule with little overlap for small singular values.

For large singular values, the Fourier transform of the visible and hidden singular vectors are again identical and they are again concentrated at low frequencies. The singular vectors associated to small singular values, are again not concentrated at low frequencies and the Fourier transform of the visible and hidden singular vectors no longer agree.

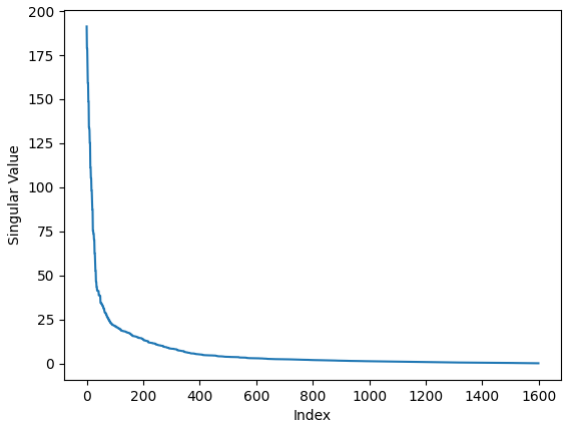


Fig. B.4 Singular values from a weight matrix generated from a coarse graining rule with large overlap.

B.6 Radial FFT

We study the frequency domain of our singular vectors using the two-dimensional discrete Fourier transform[3]

$$F(u,v) = \sum_{m=-\frac{L_v}{2}}^{\frac{L_v}{2}} \sum_{n=-\frac{L_h}{2}}^{\frac{L_h}{2}} f(v_m, h_n) e^{-i2\pi(\frac{mu}{L_v}+\frac{nv}{L_h})}. \tag{B.41}$$

Assuming that we want to learn features of the data that are rotationally invariant, it makes sense to consider the radially averaged Fourier transform. This significantly simplifies the presentation of the data and it makes comparisons easier to interpret.

After we have completed the singular value decomposition of the weight matrix, we have the hidden and visible singular vectors as one-dimensional vectors. These one-dimensional vectors are rearranged to give states of a two-dimension square lattice with $L_v \times L_v$ lattice sites for the visible vectors and $L_h \times L_h$ sites for the hidden vectors. The center of the lattice is the zero frequency and the frequency increases radially from the center. We average values in an annulus centered on the zero frequency site, to get a single radially averaged frequency mode.

[3]Note that the conventions for the Fourier transform in this section are related to those of Section B.3 by shifting the momentum space lattice.

B.7 Numerical results

In this Appendix we collect the numerical results that were briefly mentioned in the body of the article.

B.7.1 Deep RBM trained on Ising data

In Section 4.1.1, we claimed that each layer of the layered RBM performs a renormalization group transformation. The numerical evidence for this statement again comes from looking at the Fourier transform of singular vectors associated to large singular values. We again find that these singular vectors have support on low Fourier modes and that the hidden singular vector is obtained from the visible singular vector by discarding high Fourier modes.

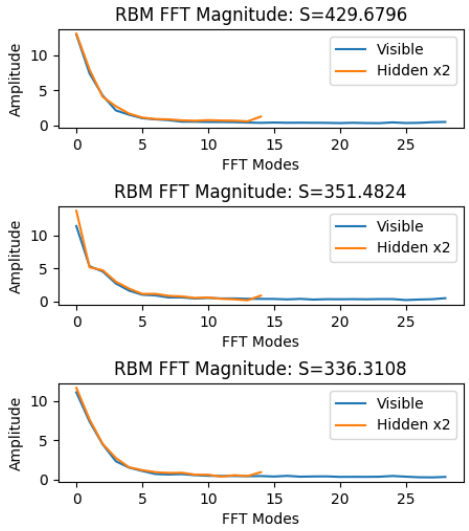


Fig. B.5 The radial Fourier transform of singular vectors of the trained weight matrix, corresponding to large singular values, for an RBM trained on Ising data. The singular vectors shown were obtained from a singular value decomposition of the weight matrix associated to the second layer of a three layer network.

Figure B.5 shows the radially averaged Fourier transform for the large singular vectors of layer 2, while Figure B.6 shows the same thing for layer 3.

B.7.2 MNIST

The MNIST data set is a rather simple data set. We have seen that training from the RGM cuts training time by a factor of 4. The fashion MNIST data set, was introduced as a more challenging data set. We have compared the performance of the RGM, a trained RBM and a trained RGM on the fashion MNIST data set. In this case, we find a speed up of 10 times.

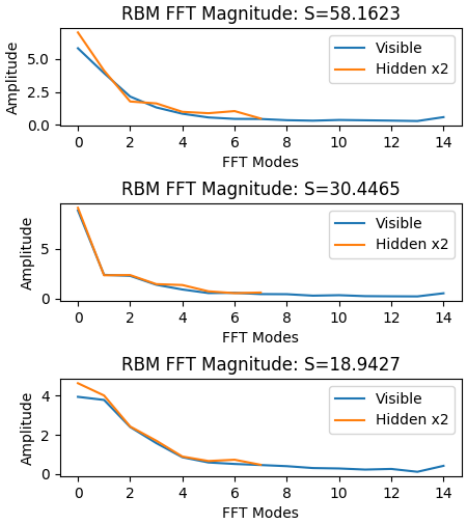


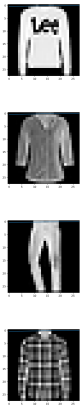
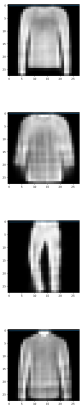
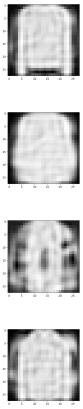
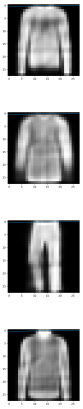
Fig. B.6 The radial Fourier transform of singular vectors of the trained weight matrix, corresponding to large singular values, for an RBM trained on Ising data. The singular vectors shown were obtained from a singular value decomposition of the weight matrix associated to the third layer of a three layer network.

Recall that the flower data set used in Section 4.1.3 was even more challenging. It lead to a speed up of about 100 times. The fashion MNIST results are shown in Table B.1.

B.7.3 Singular value decomposition of the block spin transformation coarse graining

We have used both the block spin transformation to coarse grain and the momentum space renormalization group. In this Appendix we show the equivalence of the two by performing a singular value decomposition of the matrix implementing the block spin coarse graining. The results are given in Figure B.7 and Figure B.8. The visible and hidden singular vectors have all of their support at low Fourier modes, and the hidden singular vector is again obtained from the visible singular vector by discarding large Fourier modes. In this case too, singular vectors associated to small singular values are again spread across Fourier space and the hidden and visible singular vectors again no longer agree.

Table B.1 Results of the RBM and RGM on the fashion-MNIST dataset. The RBM is trained for 1000 epochs and the RGM has been trained for 100 epochs.

Original Image	RBM	RGM	Trained RGM
			

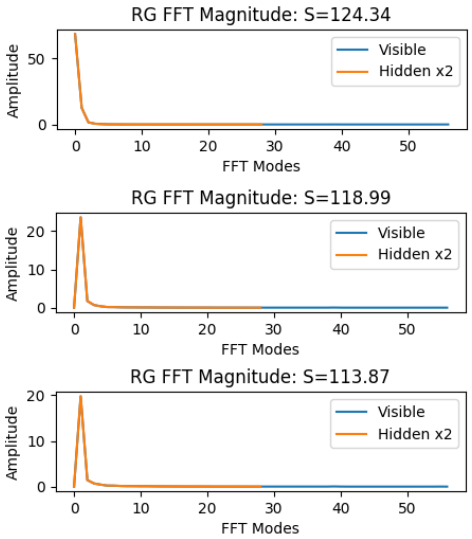


Fig. B.7 The Fourier transform of the visible and hidden singular vectors associated to the five largest singular vectors, obtained from the singular value decomposition of the block spin averaging matrix.

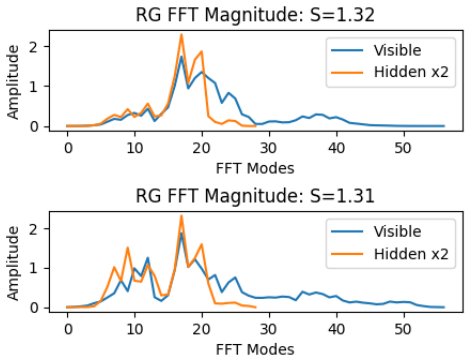


Fig. B.8 The Fourier transform of the visible and hidden singular vectors associated to the five largest singular vectors, obtained from the singular value decomposition of the block spin averaging matrix.