TRIVIALIZING THE ENERGY LANDSCAPE OF DEEP NETWORKS

Pratik Chaudhari∗ Stefano Soatto†

ABSTRACT

We study a theoretical model that connects deep learning to finding the ground state of the Hamiltonian of a spherical spin glass. Existing results motivated from statistical physics show that deep networks have a highly non-convex energy landscape with exponentially many local minima and energy barriers beyond which gradient descent algorithms cannot make progress. We leverage a technique known as topology trivialization where, upon perturbation by an external magnetic field, the energy landscape of the spin glass Hamiltonian changes dramatically from exponentially many local minima to “total trivialization”, i.e., a constant number of local minima. There also exists a transitional regime with polynomially many local minima between these two regimes. We show that a number of regularization schemes in deep learning can benefit from this phenomenon, in particular, we propose order heuristics for choosing regularization coefficients and annealing schemes for external perturbations that gradually “untrivialize” the energy landscape. We demonstrate through experiments on fully-connected and convolutional neural networks that our annealing schemes that employ trivialization lead to accelerated training and also improve generalization error.

1 INTRODUCTION

Deep learning has enjoyed enormous success in recent years in applications ranging from image recognition (Krizhevsky et al., 2012), speech recognition (Hinton et al., 2012a) and natural language processing (Bengio et al., 2003) to reinforcement learning (Mnih et al., 2015). Consequently, a number of recent papers have focused on different aspects of this problem such as the hierarchical structure (Patel et al., 2015; Salakhutdinov and Hinton, 2009; Damianou and Lawrence, 2012; Soatto, 2014), robustness of representations (Chatfield et al., 2014; Nguyen et al., 2014), large scale optimization on distributed platforms (Seide et al., 2014; Dean et al., 2012; Vasilache et al., 2014) etc. This paper is in the spirit of results that analyze algorithms for training deep networks using knowledge of the landscape of the cost function, e.g., Saxe et al. (2013); Dauphin et al. (2014); Haeffele and Vidal (2015); Janzamin et al. (2015); Choromanska et al. (2014).

We derive motivation from the theory of spin glasses which are some of the oldest models for neural networks (Hopfield, 1982). In recent years, the connections of spin glasses to combinatorial optimization, random k-SAT, LDPC codes etc. have become popular (Mezard and Montanari, 2009) and an understanding of the energy landscape and thermodynamics of such systems has led to a number of efficient algorithms (Krzakala et al., 2011; Braunstein et al., 2005). Spin glasses are particularly attractive models for large-scale problems such as deep networks because although their energy landscape is a high-dimensional, non-convex Gaussian surface, using results from probability theory and statistical physics, we can explicitly compute various quantities of interest such as the number of stationary points (Auffinger et al., 2013; Fyodorov, 2013), low temperature topology and the structure of the Gibbs distribution (Talagrand, 2003; Mezard et al., 1987).

∗ Computer Science, UCLA. pratikac@ucla.edu
† Computer Science, UCLA. soatto@cs.ucla.edu
In this work, we consider a model for deep networks with random, sparse weights and connect its loss function to the Hamiltonian of a spherical spin glass in order to analyze and modify its energy landscape. Existing results drawn from statistical physics have shown that there are exponentially many local minima and saddle points of the Hamiltonian and gradient descent algorithms cannot make progress due to such a landscape. An effective way of changing this topology is to add an external perturbation. We show that there exists a critical threshold of the perturbation below which the exponentially many stationary points persist and above which the landscape becomes “trivial”, i.e., there is only one local minimum. This phase transition is however not sharp, i.e., there exists a small band around the perturbation threshold that interpolates between the two extremes resulting in polynomially many stationary points. This polynomial regime gives a good trade-off between a complex landscape with exponentially many local minima and a completely degenerate loss function due to large perturbations.

We interpret topology trivialization as a regularization of the original loss function of a deep network, for instance, additive noise is immediately seen as a random perturbation of the Hamiltonian. We extend the idea to other types of regularization, e.g., weight decay, dropout, multiplicative noise and compute the appropriate magnitude of the regularization coefficients in the presence of perturbations to result in polynomially many stationary points of the Hamiltonian. Our analysis also motivates a very natural annealing scheme for these perturbations. The magnitude of the perturbation is gradually changed to transition from the trivialized landscape in the beginning to recover the original landscape as the training progresses. Finally, we discuss experiments that show that the annealing schemes derived from our analysis can also be applied to convolutional neural networks and fully-connected deep networks with rectified linear unit (relu) nonlinearities to result in improved training performance and generalization error.

1.1 RELATED WORK

Our work is closely related to Choromanska et al. (2014) where the authors use spin glasses to analyze the energy landscape of deep networks. They show in their experiments that although there are exponentially many saddle points in the energy landscape, the Hessian at the optima recovered by stochastic gradient descent (SGD) has very few negative eigenvalues. Choromanska et al. (2014) and Sagun et al. (2014) also show that SGD fails to progress beyond a very specific energy barrier, one that can be seen in spin glass computations as the onset of high-order saddle points and local minima. We build upon these results and demonstrate a way to use topology trivialization to modify this energy landscape so as to make it more amenable to gradient descent.

The random, sparse model discussed in Sec. 2.1 is motivated from recent results like Arora et al. (2014) and Neyshabur and Panigrahy (2013). Empirically, a number of works such as Denil et al. (2013); Chen et al. (2015) have reported that as much as 95% connections in a typical deep network have almost zero weights. Such considerations have also resulted in state-of-the-art architectures, for instance, the “inception network” (Szegedy et al., 2014) maintains a high-dimensional and sparse feature map at each layer and compresses it when convolutions of these feature maps become expensive. While a random, sparse network is unlikely to be a good model for convolutions, it allows us to make necessary approximations to get a good analytical handle on its properties. Nevertheless, our experiments in Sec. 5 show that the insight derived from trivialization on this model also applies to fully-connected and convolutional neural networks.

After the first draft of the present paper came out, we found that our work bears close resemblance to a recent paper by Neelakantan et al. (2015) where the authors employ an annealing on additive Gaussian noise to show through experiments that it helps training and generalization of complex deep networks. Our work provides a formal analysis of this phenomenon; indeed the annealing is a direct application of trivialization of the original landscape. The magnitude of noise used by Neelakantan et al. (2015) scales as $k^{-\gamma}$ for the $k^{th}$ iteration and a hyper-parameter $\gamma > 0$, while our analysis provides a more explicit formula which depends on the number of neurons $n$, number of layers $p$. 
Most surprisingly, the hyper-parameter $\gamma$ suggested by our formula, $\gamma = 0.5$, is very close to $\gamma = 0.55$ used in their experiments.

As we train deep networks on increasingly large and complex problems, hyper-parameter search is fast becoming essential (Bergstra and Bengio, 2012; Snoek et al., 2012). Values of parameters such as kernel sizes, mini-batch size, data augmentation etc. are primarily driven by hardware while others such as regularization coefficients, learning rates, gradient descent hyper-parameters (Sutskever et al., 2013; Bengio, 2012) are based on either experience or a grid-based search. Our analysis suggests the magnitude of regularization coefficients in terms of the average number of neurons on each layer and the total number of layers. This can be potentially used to accelerate hyper-parameter search in deep learning.

2 Preliminaries

2.1 A model for deep networks

Let us consider a deep network with $p$ hidden layers and $n$ neurons on each layer. Denote by $X \in \{0, 1\}^n$ the observed layer at the bottom which is generated by $h \in \{0, 1\}^n$ at the top. More precisely, $X$ can be written as

$$X = g(W^{(1)^\top} g(W^{(2)^\top} \ldots g(W^{(p)^\top} h) \ldots));$$

where $W^{(k)} \in [-1, 1]^{n \times n}$ for $k \leq p$ are the connectivity matrices and $g$ are threshold nonlinearities defined by $g(x) = 1_{x \geq 0}$. $1_A$ denotes the indicator function of the set $A$. The output of the intermediate layers is thus given by $h^{(k-1)} = g(W^{(k)^\top} g(W^{(k+1)^\top} \ldots g(W^{(p)^\top} h) \ldots))$ where $h^{(k-1)} \in \{0, 1\}^n$.

Note that, in this notation, we have $X = h^{(0)}$ and $h = h^{(p)}$. To make this model analytically tractable, we make the following assumptions:

(i) $h$ has at most $\rho$ non-zero entries,

(ii) For $d = n^{1/p}$, every entry $W_{ij}^{(k)}$ is a zero-mean random variable with $P(W_{ij}^{(k)} > 0) = P(W_{ij}^{(k)} < 0) = \frac{d}{n^2}$ and zero otherwise. This results on an average $d$ non-zero entries in every row and column.

These assumptions make our model a random, sparse neural network and help us analyze it using techniques from probability. The first assumption implies that each data sample $X_i$ is generated from a hidden feature vector $h_i$ which has at most a constant number of non-zeros, i.e., if we interpret every entry of $h_i$ as an indicator of some class, at most $\rho$ classes are present in every data sample $X_i$. The Chernoff bound then implies that with high probability, each $X_i$ has a constant $\rho (\frac{d}{n})^p$ fraction non-zero entries.

The above model enjoys a number of useful properties by virtue of it being random and sparsely connected (Arora et al., 2014). For instance, for low sparsity ($d < n^{1/5}$ or equivalently, $p > 5$), the network exhibits “weight-tying”, i.e., it acts as a denoising auto-encoder using the same weights

$$h = g(W^{(p)^\top} g(W^{(p-1)^\top} \ldots g(W^{(1)^\top} \bar{X} - \frac{d}{3} 1_n) - \frac{d}{3} 1_n) \ldots);$$

where $1_n$ denotes a vector of 1s of length $n$ and $\bar{X} = X + \delta$ for some small noise $\delta$. Equivalently, the feature vector $h$ can be obtained from $X$ by running the same network in reverse with a different thresholding function $g'(x) = 1_{x \geq \delta}$. Arora et al. (2014) also devise a community detection style algorithm to provably learn such a network. We will use the weight-tying property to compute the distribution of the energy function of such a network.

Our objective in this paper will be to use this model as an analytical tool to enable computations that connect deep networks to spin glasses. With a slightly different set of assumptions, there are other
ways of achieving this, e.g., the model in Choromanska et al. (2014) assumes that paths from each \( X_i \) to the loss layer are independent, whereas we show that the contribution of different paths starting at \( X_i \) is small due to our assumption of sparsity.

**Classification model:** In a typical classification scenario, one uses a support vector machine (SVM) or a soft-max layer on the feature vector \( h \) to obtain an output \( Y \in \{0, 1\} \). We model this as

\[
Y = g' \left( W^{(p+1)} g' \left( W^{(p)} \ldots g' \left( W^{(1)} X \right) \ldots \right) \right);
\]

where \( g'(x) = 1_{\{x \geq \frac{d}{2}\}} \) is a thresholding function and \( W^{(p+1)} \in [-1, 1]^n \) is a \( d \)-sparse row vector. The data sample \( X \in \{0, 1\}^n \) is denoted as the “input” while the output is \( Y \in \{0, 1\} \). We will henceforth consider (3) as the canonical model of a deep neural network.

### 2.2 Deep Networks as Spin Glasses

Spin glasses are used in statistical physics to analyze properties of magnetic alloys and show a number of fascinating properties due to the presence of both ferromagnetic and anti-ferromagnetic interactions. There is a wealth of literature from both physicists (Mezard et al., 1987; Mezard and Montanari, 2009) and mathematicians (Talagrand, 2003; Panchenko, 2013) that analyzes the energy landscape of spin glasses as well as their applications to optimization problems. A \( p \)-spin glass for an integer \( p \geq 1 \) with \( n \) “spins” is given by an energy function (usually called the Hamiltonian)

\[
-H_{n,p}(\sigma) = \frac{1}{n^{(p-1)/2}} \sum_{i_1, i_2, \ldots, i_p = 1}^{n} J_{i_1, \ldots, i_p} \sigma_{i_1} \ldots \sigma_{i_p},
\]

where \( \sigma = (\sigma_1, \ldots, \sigma_n) \) is a configuration of the spins and \( J_{i_1, \ldots, i_p} \) are iid zero-mean standard Gaussian random variables. Note that \( H_{n,p}(\sigma) \) is zero-mean and Gaussian as well. The constant \( n^{-(p-1)/2} \) is present to ensure that the Hamiltonian is extensive, i.e., it scales as \( \Theta(n) \). For the purpose of analysis, we will assume a spherically constrained spin glass, i.e., \( n^{-1} \sum \sigma_i^2 = 1 \), or equivalently \( \sigma \in S^{n-1}(\sqrt{n}) \subset \mathbb{R}^n \).

We now show that the loss function for the model introduced in Sec. 2.1 can be written as the Hamiltonian of a \( p \)-spin glass, please see Appendix C for the proofs. We write the output \( Y \) as a function of the “active paths” from each input neuron \( X_i \) to the output \( Y \) to get rid of the nonlinearities \( g' \). Using concentration of the weights along these paths and the independence assumptions introduced in Sec. 2.1, we obtain the following lemma.

**Lemma 1.** The output \( Y \) of the deep network in (3) can be approximated by

\[
\bar{Y}(\sigma) \overset{\text{law}}{=} \sum_{i_1, i_2, \ldots, i_p = 1}^{n} J_{i_1, \ldots, i_p} \sigma_{i_1} \ldots \sigma_{i_p},
\]

where \( J_{i_1, \ldots, i_p} \) is a zero-mean Gaussian random variable with variance \( n^{-(p-1)} \) and \( \sigma \in [-1, 1]^n \).

**Remark 2.** Note that whether the path \( i_1 \rightarrow \ldots \rightarrow i_p \) is active in the network or not is a function of both the input and the nonlinearities. This lemma uses the assumptions in Sec. 2.1 to approximate this by a Gaussian random variable \( J_{i_1, \ldots, i_p} \). Also, since the weights \( W_{i_k,i_{k+1}} \) are equally distributed and random, we can write them as simply a function of \( i_k \) which gives rise to the term \( \sigma \in [-1, 1]^n \).

We emphasize that in the sequel, \( \sigma \) is the quantity that characterizes the weights of a deep network; the expression (5) can be interpreted as the distribution of the output \( Y \) as a function of the weights \( \sigma \), where the randomness comes from both the data distribution and the weight matrices \( W \).

As the following lemma shows, for the zero-one loss defined as \( L(\bar{Y}, Y') = |\bar{Y} - Y'| \) where \( Y' \) is the true label, we can define a corresponding spin glass Hamiltonian and study its distribution.

**Lemma 3.** If the true label \( Y' \sim \text{Ber}(q) \) for some \( q < 1 \) is independent of data, the Hamiltonian defined as \( H_{n,p} = (1 - 2q)^{-1} \left( 1 - L(\bar{Y}, Y') \right) \) has distribution

\[
-H_{n,p}(\sigma) \overset{\text{law}}{=} \frac{1}{n^{(p-1)/2}} \sum_{i_1, i_2, \ldots, i_p = 1}^{n} J_{i_1, \ldots, i_p} \sigma_{i_1} \ldots \sigma_{i_p}.
\]
3 Perturbations of the Hamiltonian

In this section, we present results wherein a perturbation term is added to a general isotropic Hamiltonian and depending upon the magnitude of the perturbation, one can radically change the energy landscape of the Hamiltonian. If the perturbation is large, the landscape is dominated by the perturbation term resulting in significantly fewer stationary points, while if the magnitude is small the Hamiltonian retains its original landscape with exponentially many stationary points. This section uses some basic terminology of the Gaussian Orthogonal Ensemble (GOE) which is reviewed in Appendix B.

Let us define the total number of stationary points in the set $(-\infty,u]$ as

$$\text{crt}(u) = \sum_{\sigma: \nabla H(\sigma) = 0} 1 \left\{ n^{-1} H(\sigma) \leq u \right\}.$$  

With minor abuse of notation, denote $\text{crt}(\infty) = \text{crt}(H)$. The index of a stationary point is the number of negative eigenvalues of the Hessian of $H_{n,p}(\sigma)$ and we denote stationary points of index $k$ by $\text{crt}_k(u)$. In this notation, local minima are therefore given by $\text{crt}_0(u)$. Note that $\text{crt}_k(u)$ and $\text{crt}(u)$ are random variables but as Auffinger et al. (2013) show, we can explicitly compute their limiting values. For any $k \geq 0$ and a $p$-spin glass with $p \geq 2$ we have

$$\lim_{n \to \infty} \frac{1}{n} \log E \text{crt}_k(u) = \Theta_k(u),$$  

$$\lim_{n \to \infty} \frac{1}{n} \log E \text{crt}(u) = \Theta(u),$$

where $\Theta_k(u)$ and $\Theta(u)$ are large deviation rate functions that can be computed analytically and are shown in Fig. 1 for $p = 3$ (also see Appendix A). The notation $E$ denotes expectation, and for simplicity of notation, we skip the parentheses around it, $E$ will always apply to all the quantities to its right. The functions $\Theta_k$ and $\Theta$ are also known as the complexity of a spin glass, Fig. 1 elaborates upon them. In particular, note that the original Hamiltonian (6) has exponentially many stationary points.

### 3.1 Scaling of critical points

We introduce topology trivialization with the following theorem that characterizes the expected number of critical points in terms of the density of eigenvalues of GOE; it is a computation similar to (7) and (8).

**Theorem 4 (Fyodorov (2013)).** Consider a Hamiltonian $H(\sigma)$ where $\sigma \in S^{n-1}(\sqrt{n})$ and the covariance $E H(\sigma) H(\sigma') = F(\sigma^\top \sigma')$ is invariant under the transformations $\sigma \mapsto O\sigma$, $\sigma' \mapsto O\sigma'$ for any $n \times n$ orthogonal matrix $O$. The expected number of stationary points of $H$ is given by

$$E \text{crt}(H) = 4n \left( \frac{1 + B}{1 - B} \right)^{n/2} \sqrt{1 - B} G(B),$$

where

$$B = \frac{F''(n) - F'(n)/n}{F''(n) + F'(n)/n},$$

$$G(B) = \int_0^\infty e^{-nBt^2/2} \rho_n(t) \, dt,$$

and $\rho_n(t)$ is the density of the eigenvalues of GOE.
Now consider a perturbed version of the $p$-spin glass Hamiltonian in (6) given by

$$-\tilde{H}(\sigma) = \sum_{i_1,\ldots,i_p=1}^{n} J_{i_1,\ldots,i_p} \sigma_{i_1} \cdots \sigma_{i_p} + \sum_{i=1}^{n} h_i \sigma_i; \quad (9)$$

where (as before) $J_{i_1,\ldots,i_p}$ is a zero-mean Gaussian random coupling with variance $J^2/p^{p-1}$ and we have added an external magnetic field given by $\sum_i h_i \sigma_i$. This is a random external field, i.e., $h_i$ are iid zero-mean Gaussian random variables with variance $E h_i^2 = v^2$. An application of Thm. 4 now shows that the parameter $v$ results in three distinct regimes for the expected number of critical points. First note that $H(\sigma)$ satisfies the isotropic covariance of Thm. 4 with

$$E \tilde{H}(\sigma) \tilde{H}(\sigma') = n f \left( \frac{\sigma^\top \sigma'}{n} \right), \quad f(u) = J^2 u^p + v^2 u;$$

and $B$ can be calculated to be

$$B = \frac{J^2 p(p-2) - v^2}{J^2 p^2 + v^2} \in \left( -1, \frac{p-2}{p} \right]. \quad (10)$$

The parameter $B$ characterizes the trivialization phenomenon completely, we formalize this as the following theorem.

**Theorem 5 (Fyodorov (2013)).** For $p > 2$, as $n \to \infty$, if $B > 0$ there exist exponentially many critical points, if $B = -\mathcal{O}(n^{-1})$ there exist polynomially many critical points while if $B = -\Omega(n^{-1})$ the number of critical points of $\tilde{H}$ is exactly two, in expectation. More precisely,

$$\lim_{n \to \infty} E \text{crt}(\tilde{H}) = \begin{cases} \frac{2}{\sqrt{n}} \tau^{-3/2} \exp \left( \frac{\tau}{2} \log \frac{1+\beta}{1-\beta} \right) & \text{if } B = -\Omega(n^{-1}), \\ 4n^{1/2} \sqrt{1+\beta} \exp \left( \frac{\tau}{2} \log \frac{1+\beta}{1-\beta} \right) & \text{if } B > 0. \end{cases} \quad (11)$$

where $B = 0$ corresponds to $\nu_c = J \sqrt{p(p-2)}$ and $\tau \gg 1$ is any constant that does not scale with $n$.

The first phase in the above theorem demonstrates total trivialization of the energy landscape, indeed as any smooth function on the sphere $S^{n-1}(\sqrt{n})$ has at least one minimum and at least one maximum, there is in fact exactly one minimum if the perturbations to the Hamiltonian are larger than the critical value $J \sqrt{p(p-2)}$. If the perturbations are smaller than this, we are back to the regime of exponentially many critical points in (8), as shown in the third case. There is also an interesting “edge-scaling” regime around the value $B = 0$ where if $B = -\mathcal{O}(1/n)$ the spin glass has polynomially many — more precisely, $\mathcal{O}(n)$ — critical points. This is the regime where we would typically like to choose our regularization parameters to be, to not only avoid exponentially many critical points, but also avoid total trivialization. Finally, let us remark that the number of local minima is at most the number of critical points. Also, Thm. 5 only describes the behavior of the expected number of critical points, however as Subag (2015) show, $\lim_{n \to \infty} \frac{\text{crt}(u)}{E \text{crt}(u)} = 1$ in $\mathbb{L}^2$ and thus the number of critical points converges to the expected number of critical points in probability.

**Corollary 6.** For $p \gg 2$, as $n \to \infty$, we need $\nu = \mathcal{O}(p(1 + 2\tau/n))$ to be in the polynomial regime given by Thm 5. In particular, for the model considered in Sec. 2.1 where $p = \log_d n$, we require $\nu = \mathcal{O}(\log_d n (1 + 2\tau/n))$ to be in the polynomial regime.

The above corollary follows directly from (10) and the fact that Thm. 5 requires $B = -\tau/n$ for the polynomial regime. We now use the following lemma to make the estimate in polynomial regime more precise.

**Lemma 7 (Fyodorov (2013)).** If $B = -\tau/n$, $\tau \in \mathbb{R}$ the expected number of critical points scales as

$$\lim_{n \to \infty} \frac{E \text{crt}(\tilde{H})}{2n} = -2 \frac{d}{d\tau} \left[ 2^{-\gamma/2} I_0 \left( \frac{\tau}{2} \right) \right];$$

where $I_0(z) = \pi^{-1} \int_0^\pi \exp(z \cos \phi) \, d\phi$ is the modified Bessel function. In particular, we recover

$$\lim_{n \to \infty} \frac{E \text{crt}(\tilde{H})}{2n} = \begin{cases} \frac{1}{\sqrt{\pi |\tau|}} \tau^{-3/2} & \text{for } \tau \gg 1, \\ \frac{1}{\sqrt{\pi |\tau|}} 2e^{\tau} & \text{as } \tau \to -\infty. \end{cases} \quad (12)$$
We now interpret different types of regularization in deep networks as perturbations of the original loss function in (6) and compute the scaling of regularization coefficients in terms of the number of neurons $n$ and the number of layers $p$.

### 4.1 ADDITIVE NOISE

The weight update equation in the presence of additive noise (Jim et al., 1996) can be written as

$$\sigma \leftarrow \sigma - \eta \nabla H_{n,p}(\sigma) - \Delta$$
where $\Delta \in \mathbb{R}^n$ is a zero-mean Gaussian random vector with variance $\nu^2 I_{n \times n}$. $\nabla H_{n,p}(\sigma)$ is the back-propagated gradient and $\eta$ is the learning rate. Additive noise is typically used to escape local minima and thus it is important to pick the right magnitude of var $\Delta$ to avoid large jumps in the weights $\sigma$ at every iteration that are uncorrelated with the gradient. The perturbed Hamiltonian which gives the weight update equation above is $-\bar{H}_{\text{add}}(\sigma) = -H_{n,p}(\sigma) + \sum_{i=1}^{n} \Delta_i \sigma_i$. This is the same as our original perturbed Hamiltonian (9) and for a constant $\tau$, we have from Cor. 6 that

$$v = Jp \left(1 + \frac{2\tau}{n}\right)^{1/2}.$$  

4.2 Weight decay

Weight decay (Moody et al., 1995) implies an $\ell_2$ regularization on the weights or equivalently, a Gaussian prior on them. We write the weight updates as

$$\sigma \leftarrow \sigma - \eta \nabla H_{n,p}(\sigma) - \frac{p \alpha}{\sqrt{n}} \sigma$$

for a constant $\alpha > 0$; the factor of $pn^{-1/2}$ is merely for convenience. The decay term above can be obtained using an additive term of the form $\frac{p \alpha}{2\sqrt{n}} \sum_{i=1}^{n} \sigma_i^2$ in the Hamiltonian; this is akin to introducing a spherical constraint on the spins weighted by a Lagrange multiplier $p \alpha / (2\sqrt{n})$ and hence will not lead any phase transitions in the number of critical points. However, if we add a perturbation term in addition to weight decay, we can expect similar behavior as Sec. 3. Consider the zero-mean perturbation

$$\bar{H}_{\text{decay}}(\sigma) = -H_{n,p}(\sigma) + \frac{p\alpha}{2\sqrt{n}} \sum_{i=1}^{n} \sigma_i^2 + \sum_{i} h_i \sigma_i.$$  

(15)

with var $h_i = \nu^2$ which creates an additive term of $h$ in the update rule along with the weight decay term. $\bar{H}_{\text{decay}}$ is now amenable to the analysis of Thm. 5 with the covariance given by

$$E \bar{H}_{\text{decay}}(\sigma) \bar{H}_{\text{decay}}(\sigma') = \frac{J^2}{np} \sum_{i_1, \ldots, i_p} \sigma_{i_1} \sigma_{i_1}' \ldots \sigma_{i_p} \sigma_{i_p}' + \frac{p^2 \alpha^2}{4} \sum_{i,j} (\sigma_i \sigma_j)'^2 + \nu^2 \sum \sigma_i \sigma_i';$$

$$\approx n \left(J^2 u^p + \frac{p^2 \alpha^2}{4}u^2 + \nu^2 u\right), \quad \text{where } u = \frac{\sigma^T \sigma}{n};$$

$$\Rightarrow \eta f \left(\frac{\sigma^T \sigma}{n}\right).$$

The approximation in the second step is obtained by assuming that $E\sigma_i \sigma_i' \approx 0$ for all $i \neq j$ which roughly says, averaged over the disorder $J$, two deep networks given by $\sigma, \sigma'$ have uncorrelated $i^{th}$ and $j^{th}$ neurons on each layer. We now use the above expression to calculate the value of $B$ in (10) to get $B = \frac{J^2 p(p-2) - \nu^2}{p^2 + p\alpha^2 + \nu^2}$. Note that the value of $\nu$, which corresponds to $B = 0$ is unchanged from Thm. 5, but to stay in the polynomial regime, if we set $B = -\tau n^{-1}$ we now have, for large $n$,

$$v = J p \left(1 + \tau \frac{\alpha^2}{n} + \frac{2\tau}{n}\right)^{1/2}.$$  

(16)

Contrast this with Cor. 6 or additive noise in Sec. 4.1, the strength of the perturbation needs to be larger to account for the curvature of the $\ell_2$ regularization term.

4.3 Multiplicative noise

If instead of a uniform constant, we use a random matrix $\Delta \in \mathbb{R}^{n \times n}$ (usually diagonal) with iid entries $E \Delta_{ii} = 0$ and $E \Delta_{ii}^2 = \alpha^2$ to multiply the weights, we get multiplicative noise (Nalisnick et al., 2015) with the update rule

$$\sigma \leftarrow \sigma - \eta \nabla H_{n,p}(\sigma) - \frac{p}{\sqrt{n}} \Delta \sigma.$$
Similar to weight decay, we add a perturbation term to the corresponding Hamiltonian to get
\[-\tilde{H}_{\text{mult}}(\sigma) = -H_{a,p}(\sigma) + \frac{p}{2\sqrt{n}} \sum_{i=1}^{n} \Delta_i \sigma_i^2 + \sum_i h_i \sigma_i.\]  
(17)
The analysis follows that of Sec. 4.2 and we get the same expression as (16) although since entries of \(\Delta\) are iid, we need not make the approximation \(E\sigma_i \sigma'_j \approx 0\) for all \(i \neq j\).

4.4 DROP-OUT AND DROP-CONNECT

Drop-Out (Hinton et al., 2012b) is a form of regularization typically used for fully connected layers where each output is forwarded with a fixed probability \(q\) and set to zero otherwise. Numerous experiments have shown that drop-out leads to a better generalization error for deep networks by forcing each layer to explain the layer above using as few features as possible. Drop-Connect (Wan et al., 2013) is very similar to drop-out except that one sets the individual weights to zero with probability \(1-q\). This induces sparsity in the weight matrix at each layer. We model drop-connect by introducing a Bernoulli random variable \(\delta_1 \ldots \delta_p\) that multiplies the disorder \(J_{i_1 \ldots i_p}\) to give
\[-\tilde{H}_{\text{drop-connect}}(\sigma) = \sum_{i_1,i_2,\ldots,i_p} \delta_{i_1} \ldots \delta_{i_p} J_{i_1 \ldots i_p} \sigma_{i_1} \ldots \sigma_{i_p} + \sum_i h_i \sigma_i\]  
(18)
where as before \(J_{i_1 \ldots i_p}\) is zero-mean Gaussian with variance \(J^2 n^{-(p-1)}\). Such a Hamiltonian is known as a “diluted” spin glass. Since \(\delta_1 \ldots \delta_p = 0\) is like setting the entire path \((i_1,\ldots,i_p)\) to zero, we set \(\delta_1 \ldots \delta_p \sim \text{Ber}(q^p)\). Similar to the proof of Lem. 1, we now replace \(\delta_1 \ldots \delta_p J_{i_1 \ldots i_p}\) by a Gaussian random variable to put it into the framework of Sec. 3. If we denote \(c \triangleq q^p\), this amounts to scaling the constant \(J\) by \(\sqrt{c(1-c)}\) which gives \(\text{var} J_{i_1 \ldots i_p} = n^{- (p-1)} c(1-c) J^2\). For the perturbation term to result in polynomially many critical points, we now require
\[v = J p \sqrt{c(1-c)} \left(1 + \frac{2\tau}{n}\right)^{1/2}.\]  
(19)

Quite intuitively, since \(c(1-c) < 1/2\), drop-connect requires that the magnitude of the perturbation term be reduced by an appropriate factor. As a minor aside, following the idea that drop-connect sets entire “paths” from data layer to loss layer to zero we should pick the drop-out probability such that \(q^p\) is a constant, i.e., \(q = c^{1/p}\).

To model drop-out, we multiply each spin in (9) by \(\delta_i \sim \text{Ber}(q)\) to get
\[-\tilde{H}_{\text{drop-out}}(\sigma) = \sum_{i_1,i_2,\ldots,i_p} J_{i_1 \ldots i_p} \prod_{k=1}^{p} \delta_{i_k} \sigma_{i_k} + \sum_i h_i \sigma_i\]  
(20)
For our model, the connectivity matrices \(J_{i_1 \ldots i_p}\) are such that each neuron \(i_k\) is connected to \(i_{k+1}\) in the next layer uniformly randomly. This enables us to interpret the factor \(\prod_k \delta_{i_k}\) as a path \(i_1 \rightarrow \ldots \rightarrow i_p\) similar to (18) and gives the same expression for the perturbation as (19).

5 EXPERIMENTS

5.1 SPIN GLASSES

In this section, we discuss simulations of a 3-spin glass to demonstrate topology trivialization. In our notation, this is a fully-connected neural network with three hidden layers, threshold nonlinearities and a fourth loss layer. Consider the perturbed Hamiltonian (4) given by
\[-\tilde{H}_{n,3} = \frac{1}{n} \sum_{i,j,k=1}^{n} J_{i,j,k} \sigma_i \sigma_j \sigma_k + \sum_i h_i \sigma_i;\]  
(21)
with \(h_i \in N(0, \nu^2)\) and \(\sigma \in S^{n-1}(\sqrt{n})\). The gradient can be computed to be
\[-\nabla \tilde{H}_{n,3} = \frac{1}{n} \sum_{j,k} \left[ J_{ijk} \sigma_j \sigma_k + J_{jik} \sigma_j \sigma_k + J_{jki} \sigma_j \sigma_k \right] + h.\]
For a given disorder $J$ we find the ground state of the above Hamiltonian for values of $h$ in the three regimes considered in this paper (cf. Cor. 6). $v = 1/n$ for exponentially many stationary points, $v = (1 + \tau/n)^{1/2}$ with $\tau \approx n/2$ to be in the polynomial regime and $v = p$ to be in the totally trivialized regime with a single minima. For an initial configuration $\sigma^1 \in S^{n-1}(\sqrt{n})$, we perform gradient descent with a fixed step size $\eta = 0.1$ and stop when the gradient $\|\nabla H\|_2 \leq 10^{-4}$. At each step, we need to project the gradient on the tangent plane of $S^{n-1}(\sqrt{n})$ and re-project $\sigma^{k+1} \leftarrow \sigma^k - \eta \nabla H(\sigma^k)$ back on the sphere. Note that normalizing an isotropic Gaussian vector makes it uniform on the unit sphere and hence we can uniformly sample $S^{n-1}(\sqrt{n})$ for these experiments easily.

In order to visualize the energy landscape, we perform a t-SNE projection (Van der Maaten and Hinton, 2008) of the local minima obtained by starting gradient descent from $N = 20,000$ uniformly sampled points of $S^{n-1}(\sqrt{n})$ for $n = 100$. t-SNE is a nonlinear dimensionality reduction technique

(a) Exponential regime: In addition to deep valleys surrounded by high energy barriers (seen as isolated dots), we also see “canyons”, i.e., long narrow passages with very similar values of the loss function. This suggests that there are large regions where the gradient is very small in all but a few directions, in other words, these are saddle points.

(b) Polynomial regime: As Thm. 5 predicts, the number of stationary points (which are clusters in the t-SNE projection) is significantly reduced as compared to Fig. 2a. Surprisingly, the landscape is full of saddle points in this regime. An investigation into the nature of the critical points resulting from trivialization in terms of the eigenvalues of the Hessian is a subject of future work.

(c) Total trivialization: All executions of gradient descent converge at the same local minimum, indeed, they have the almost same value of normalized Hamiltonian ($-3.4463$). The average distance between two points in the original high-dimensional data in Fig. 2c is 0.02 as compared to 1.16 for Fig. 2a, t-SNE thus projects all points into one large cluster which is as an indication of a unique local minimum of the Hamiltonian.

(d) Probability density of the normalized Hamiltonian in the exponential regime (same data as Fig. 2a): The external magnetic field introduces a slight positive skew in the histogram, i.e., gradient descent is more likely to converge to minima with a lower energy with trivialization. Without any trivialization, one expects (e.g., Fig. 2 in Sagun et al. (2014)) a Gaussian centered in the region $[-E_0, -E_{\text{inf}}]$ (cf. Fig. 1).

Figure 2: A two-dimensional t-SNE projection (based on Euclidean metric) of 20,000 local minima on the sphere $S^{n-1}(\sqrt{n})$ obtained by gradient descent on a 3-spin glass with $n = 100$ in the three regimes, exponential (Fig. 2a), polynomial (Fig. 2b) and trivial (Fig. 2c).
that maps neighboring points in the original high-dimensional space to close by points in the 2D projection and maps far away points to distant points in the projection. It is widely used to visualize clusters in high-dimensional data Van der Maaten and Hinton (2008). We use the Euclidean metric for points in $\mathbb{S}^{n-1}(\sqrt{n})$ and $\mathbb{R}^2$ as a measure of similarity for t-SNE in the original and projected spaces, respectively. In particular, note that we do not use the values of the Hamiltonian at these local minima to cluster. Fig. 2 discusses the results of this simulation.

5.2 Fully-connected deep networks

After demonstrating topology trivialization on spin glasses, we now employ its interpretation as a regularization procedure to train fully-connected deep networks for image classification on two datasets, viz., the MNIST dataset (LeCun et al., 1998) with $32 \times 32$ gray-scale images of digits, and the CIFAR-10 dataset (Krizhevsky, 2009) with $32 \times 32$ RGB images. We whiten both datasets by subtracting the mean and dividing by the per-pixel standard deviation, no data augmentation is performed. We use two fully-connected networks of $p = 20$ layers with relu nonlinearities and soft-max over 10 classes as a classifier. The mnistfc network has 64 hidden units on each layer while cifarfc network has 128 hidden units on each layer. Note that the model in Sec. 2.1 has $p + 1$ layers including the final soft-max layer. We denote these architectures by

\[
\text{mnistfc} : \text{input} \rightarrow \text{linear}_{64} \rightarrow \text{linear}_{10} \rightarrow \text{soft-max}, \quad (22)
\]

\[
\text{cifarfc} : \text{input} \rightarrow \text{linear}_{128} \rightarrow \text{linear}_{10} \rightarrow \text{soft-max}; \quad (23)
\]

where linear$_{64}$ indicates a fully-connected layer with 64 outputs and relu nonlinearities.

We estimate the average number of neurons on each layer $n$ for our formulas in Sec. 4 as $n = \sqrt{\#\text{weights}}/p$, this estimate is accurate if all the layers were fully-connected and of the same size. During training, introducing a magnetic field $h$ causes the weights to align with $h$ while their magnitude grows unbounded. Hence, we add a small $\ell_2$ regularization term with $\alpha = 10^{-8}$ (with the normalization in (15) in Sec. 4.2) which induces a spherical constraint to prevent this. We use a momentum-based optimization algorithm ADAM (Kingma and Ba, 2014) after tuning the initial learning rate to $\eta = 10^{-4}$ that decays by a factor of 0.95 after every epoch.

5.2.1 Annealing procedure

Let us now discuss the annealing procedure that leverages trivialization. We focus on weight decay (Sec. 4.2) for the experiments in the sequel, annealing schemes for other types of regularization can be derived similarly and results are qualitatively similar. As (16) shows, the magnitude of perturbation for weight decay is given by

\[
\nu = Jp \left(1 + \tau \frac{\alpha^2}{n} + \frac{2\tau}{n}\right)^{1/2}.
\]

Lem. 7 states that using the parameter $\tau$, we can smoothly interpolate between the polynomial regime ($\tau \gg 1$) and the original energy landscape with exponentially many critical points ($\tau \rightarrow -\infty$). This motivates an annealing scheme that decreases $\tau$ gradually as training progresses, and “untrivializes” the energy landscape. This enables gradient descent to make progress quickly in the beginning of training in the relatively benign energy landscape with only $O(n)$ critical points. As we approach the end of the training we would like to converge to the original energy landscape so as not to affect the generalization properties of the loss function. If $N$ is the total number of epochs being trained for, and $k$ is the current epoch, we set

\[
\tau(k) = -\left(\frac{n}{\alpha^2 + 2}\right) \frac{k}{N}.
\]

Note that $\tau(0) = 0$ which gives $\nu(k) = Jp$ whereas for the $N^{th}$ epoch, we have $\nu(N) = 0$. The perturbation $\nu$ thus scales as $O\left(1 + k/N\right)^{-1/2}$ if $k \ll N$. In fact, the exponent of $-1/2$ matches
very well with empirical results for an additive Gaussian noise annealing scheme in Neelakantan 
et al. (2015). In practice, we treat the constant $J$ as a tunable parameter and set its value using 
cross-validation; it is in the range $10^{-5} - 10^{-6}$ for the networks considered here. In fact, due to the 
appropriate normalization used in the equations in Sec. 4, we observe that parameters such as $\alpha$ and 
$J$ (cf. (24)) are almost constant across different networks.

Let us emphasize here that the perturbations considered in this paper — and motivated from the 
phenomenon of trivialization — are fixed at the beginning of the training as opposed to regularizations 
in the deep learning literature such as additive noise where one samples a realization of noise at 
each iteration. After every epoch, we simply rescale the magnetic field $h$ to have the magnitude 
given by (24). The motivation behind resampling, say additive gradient noise, at each iteration is 
to enable gradient descent to jump out of small local minima. In this work, we instead rely on the 
weights aligning themselves in the direction of the magnetic field in the beginning of training. As the 
magnitude of this external field reduces, the properties of stochastic gradient descent dominate the 
training performance (cf. Figs. 4b and 4c).

Figure 3: Figs. 3b and 3c show the validation error averaged over 10 runs of 50 epochs and a batch size of 
1024 for the mnistfc (22) and cifarfc (23) networks. Fig. 3a shows the training error for mnistfc; the training 
error for cifarfc is similar. The blue curves denote an $\ell_2$ regularization with $\alpha = 10^{-8}$ without trivialization 
while the red curves are obtained using the same $\ell_2$ regularization but with the noise annealing in (24) for 
$J = 10^{-5}$. Trivialization helps obtain a better generalization in a dramatic fashion, the network is very close to 
its final validation error by the 5th epoch for both mnistfc (4.9%) and cifarfc (59.8%). Note that these networks 
lack convolutions and hence we do not expect state-of-the-art validation errors. Initialization of weights in 
deep fully-connected networks is often difficult, trivialization helps in such cases, for instance, the validation 
error without trivialization in Fig. 3c remains at 90% (same as guessing uniformly) for almost 9 epochs before 
decreasing.

5.3 CONVOLUTIONAL NEURAL NETWORKS

Next, we train a CNN with the following architecture on the CIFAR-10 dataset —

$$
cifarconv : \text{input} \rightarrow \text{conv}_{5\times5\times8} \rightarrow \text{linear}_{128} \rightarrow \text{linear}_{10} \rightarrow \text{soft-max};$$

with a max-pool$_{3\times3}$ layer after each convolutional layer and relu nonlinearities after every layer. The 
notation conv$_{5\times5\times8}$ denotes a convolutional layer with a $5 \times 5$ kernel and 8 output features. We use 
appropriate zero-padding on convolutional and max-pooling layers to preserve the size of the input 
image through the convolutional part of the network. We set the total number of layers for cifarconv 
to be $p = 16$ and again estimate the number of hidden units on each layer as $n = \sqrt{\text{#weights}/p}$. 
Note that each convolutional layer has $32 \times 32 \times 8$ outputs, but has much fewer weights, precisely, 
$8 \times 25 \times 8$. The annealing procedure for the experiments using CNNs is the same as the one described 
in Sec. 5.2.1. Fig. 4a and 4b show the results for $\ell_2$ regularization and trivialization for this network.
Figure 4: Figs. 4a and 4b show the training and validation error of the cifarconv network averaged over 5 runs of 100 epochs with a batch size of 256. The blue curve denotes error with $\ell_2$ regularization and $\alpha = 10^{-8}$ while the red curve is the error with the same $\ell_2$ regularization and the annealing in (24) for $J = 10^{-5}$. The green curve denotes the same $\ell_2$ regularization and annealing but where the magnetic field $h$ is resampled at each iteration. Similar to Fig. 3c, trivialization accelerates the training in the beginning as the weights align themselves with the large external magnetic field. This can be seen as a sharp increase in the magnitude of the regularization term $\sum h_i \sigma_i$ in Fig. 4c. As the magnetic field $h$ reduces, $|\sum h_i \sigma_i|$ decreases towards zero. The effect of trivialization is less pronounced for the cifarconv network than, say Fig. 3c, probably due to the convolution operations which our model does not consider. Fig. 4d shows the magnitude of the smallest entry in the gradient for the two cases. Throughout training, trivialization results in a large non-zero gradient, which explains the increased performance. Note that resampling the magnetic field (equivalently, additive noise) at each iteration performs worse than trivialization without resampling. In particular, the gradient is bounded away from zero (cf. Fig. 4d) but because the weights are constantly forced to align with the resampled magnetic field (cf. Fig. 4c), the training and generalization performance do not improve.

The above experiments show that trivialization accelerates training and improves generalization error on “very deep” deep networks; we now discuss the results on a network with a large number of convolutional hidden units but smaller depth. We consider the network-in-network architecture (Lin et al., 2013) (cifarnin) on the CIFAR-10 dataset denoted by

$$
cifarlin: \text{input} \rightarrow \text{conv}_{5 \times 5 \times 192} \rightarrow \text{max-pool}_{3 \times 3} \rightarrow \text{dropout} \ldots \rightarrow \text{conv}_{3 \times 3 \times 192} \rightarrow \text{conv}_{1 \times 1 \times 192} \rightarrow \text{conv}_{1 \times 1 \times 10} \rightarrow \text{mean-pool}_{8 \times 8} \rightarrow \text{soft-max}.
$$

This network employs a number of regularization techniques that are designed to avoid the over-fitting introduced by large fully-connected layers. For instance, every convolutional layer above is
followed by batch-normalization (Ioffe and Szegedy, 2015) and relu nonlinearities. cifarnin also employs global average pooling to replace fully-connected layers by averages of the feature maps. As experiments in Ioffe and Szegedy (2015) and Lin et al. (2013) show, batch-normalization and global average pooling act as good regularizers and have resulted in state-of-the-art performance on a number of datasets. We treat each convolutional layer as one layer of our theoretical model and set $p = 8$ with $n = \sqrt{\text{#weights}/p}$ as before. Figs. 5a and 5b show the results of our annealing procedure while training the cifarnin network.

![Figure 5](image.png)

**Figure 5:** Figs. 5a and 5b use an $\ell_2$ regularization of $\alpha = 10^{-8}$ and are averaged over 5 runs of 100 epochs with a batch size of 32. The red curve uses the annealing procedure in (24) with $J = 10^{-6}$, while the green curve uses the same $\ell_2$ regularization and annealing but resamples the magnetic field $h$ (additive noise) at every iteration. The heavy regularization introduced by the network-in-network architecture results in smaller gains due to trivialization; the blue curve has an average testing error of 21.89% as opposed to 21.02% average error on the red curve with trivialization and 22.05% average error on the green curve (note that we do not employ data-augmentation here). Let us emphasize that the theoretical model in Sec. 2.1 from which the annealing scheme in (24) is derived, only consists of sparse, fully-connected layers with threshold nonlinearities and hence does not account for the regularizations introduced in cifarnin; nonetheless, we obtain an improved generalization error with trivialization. Resampling additive noise at each iteration (green curve) is akin to a magnetic field in a different (random) direction at each iteration and similar to Fig. 4, it performs worse than trivialization.

## 6 Conclusions

We employed results from spin glass theory to introduce a novel topology trivialization procedure for interpreting regularization techniques in deep learning. While the original energy landscape is complex with exponentially many local minima and saddle points, we showed that it undergoes dramatic changes in structure upon adding small perturbations. We can compute the threshold for the magnitude of perturbations below which the landscape is qualitatively unchanged and above which it becomes trivial. There also exists a critical band for the perturbation which results in polynomially many stationary points of the Hamiltonian and interpolates between the exponential regime and the totally trivial one.

We motivate an annealing scheme wherein the loss function can be gradually modified from the polynomial regime to the original exponential regime during the course of training. Our analysis provides order estimates for choosing regularization coefficients of a number of different regularization methods in the deep learning literature to leverage topology trivialization. These insights derived from a formal analysis are also applicable to fully-connected deep networks with nonlinearities and convolutional neural networks used in practice. We show through experiments that leveraging trivialization results in improved training performance as well as smaller generalization error.
Our experiments suggest that trivialization works well on deep networks. We also expect it to work well on other deep architectures such as recurrent neural networks (RNNs) and long short-term memory networks (LSTMs). However, popular techniques in computer vision such as convolutions, max-pooling, batch-normalization etc. already employ heavy regularization and hence — as seen in our experiments — the benefits of trivialization on these networks are smaller.

REFERENCES


### A ENERGY LANDSCAPE OF DEEP NETWORKS

#### Theorem 11 (Thm 2.5 in *Auffinger et al. (2013)*). For all \( p \geq 2 \) and \( k \geq 0 \) fixed

\[
\lim_{n \to \infty} \frac{1}{n} \log E_{n} \Theta_{k}(u) = \Theta_{k}(u);
\]

where \( \Theta_{k}(u) \) is a large deviation rate function given by

\[
\Theta_{k}(u) = \begin{cases} 
\frac{1}{2} \log(p - 1) - \frac{p - 2}{2(p - 1)} u^2 - (k + 1) I_{1}(u), & \text{if } u \leq -E_{\infty}, \\
\frac{1}{2} \log(p - 1) - \frac{p - 2}{p}, & \text{else};
\end{cases}
\]

where \( E_{\infty} = 2 \sqrt{\frac{p - 1}{p}} \) and \( I_{1} : (-\infty, -E_{\infty}] \to \mathbb{R} \) is given by

\[
I_{1}(u) = \frac{2}{E_{\infty}^{2}} \int_{u}^{-E_{\infty}} (z^{2} - E_{\infty}^{2})^{1/2} dz.
\]

#### Theorem 12 (Thm. 2.8 in *Auffinger et al. (2013)*). For all \( p \geq 2 \),

\[
\lim_{n \to \infty} \frac{1}{n} \log E_{n} \Theta(u) = \Theta(u);
\]

\[
\Theta(u) = \begin{cases} 
\frac{1}{2} \log(p - 1) - \frac{p - 2}{2(p - 1)} u^2 - I_{1}(u), & \text{if } u \leq -E_{\infty}, \\
\frac{1}{2} \log(p - 1) - \frac{p - 2}{2(p - 1)} u^2, & \text{if } -E_{\infty} \leq u \leq 0, \\
\frac{1}{2} \log(p - 1), & \text{else}.
\end{cases}
\]

We do not make use of the following fact, but as Thms. 2.14 and 2.15 in *Auffinger et al. (2013)* prove, the energy landscape shows a very intricate structure. For \( \sigma \in S^{n-1} \sqrt{n} \), if we define the lower bound of the Hamiltonian as \( \lim_{n \to \infty} n^{-1} H(\sigma) = -E_{0} \), as one progresses from the global minimum at energy level \( -nE_{0} \), saddle points of increasing indices start appearing, in particular, between energy levels \( (-nE_{0}, -nE_{k}) \), only critical points of indices less than \( k \) exist (in the limit as \( n \to \infty \)). Most importantly, note that there are exponentially many local minima in any non-zero measure interval between \( (-nE_{0}, -nE_{k}) \). This fact is crucial to construct algorithms that rely on gradient
descent, for instance, stochastic gradient descent can get stuck in any one of these exponentially many local minima and indeed, as experiments in Choromanska et al. (2014); Sagun et al. (2014) show, the minima obtained by SGD are always low order critical points, i.e., they are either local minima which could be very close to the energy barrier or they are saddle points very deep in the energy landscape. Empirically, SGD cannot penetrate the energy barrier $-nE_{\text{nc}}$ below which the layered structure presents itself. The main motivation of this paper is to add perturbations to the Hamiltonian to reduce the number of local minima and saddle points.

B  Kac-Rice formula and GOE

Given a smooth, random function $H(\sigma)$ of $n$ variables $\sigma = (\sigma_1, \ldots, \sigma_n)$, if $N_i(D)$ is the total number of stationary points in a subset $D \subset \mathbb{R}^n$, let $\rho_i(\sigma)$ be their corresponding density. We thus have $N_i(D) = \int_D \rho_i(\sigma)d\sigma$. The Kac-Rice formula (Adler and Taylor, 2009) then gives

$$\mathbb{E}\rho_i(\sigma) = \mathbb{E}\left\{ |\det (\partial^2_{k_1,k_2} V) | \prod_{k=1}^n \delta(\partial_i H) \right\}.$$  

(26)

Similarly, if one is interested only in the number of local minima, we have

$$\mathbb{E}\rho_i(\sigma) = \mathbb{E}\left\{ \det (\partial^2_{k_1,k_2} H) \theta (\partial^2_{k_1,k_2} H) \prod_{k=1}^n \delta(\partial_i H) \right\}.$$  

where $\theta (\partial^2_{k_1,k_2} H)$ is the Heavyside step function. We consider isotropic Hamiltonians with a covariance given by

$$E H(\sigma)H(\sigma') = F(\sigma^\top \sigma')$$

for some functions $F$. Note that the covariance is invariant to changes $\sigma \rightarrow O\sigma$ for any unitary, orthogonal matrix $O$. We also impose a spherical constraint $\|\sigma\|_2 = \sqrt{n}$ to match with the model in Sec. 2.1.

Using the Kac-Rice formula, we can completely characterize the expected number of critical points and local minima of isotropic Hamiltonians. The Hamiltonians we consider are high-dimensional zero-mean Gaussian surfaces and hence such an analysis is tied to the properties of random matrices with Gaussian entries which we describe briefly. (see e.g., Mehta (2004) for an elaborate exposition). A real symmetric matrix $G \in \mathbb{R}^{n \times n}$ is said to belong to the Gaussian Orthogonal Ensemble (GOE) if every entry $G_{ij}$ for $i \leq j$ is an iid zero-mean Gaussian random variable with $E G_{ij}^2 = \frac{1+\delta_{ij}}{n}$. The density of GOE matrix is proportional to $\exp\left(-\frac{1}{2} \text{tr} \, G^2\right)$ and is hence invariant to orthogonal conjugation.

The joint eigenvalue distribution of such matrices is a well-studied object, it is usually characterized as the density $\rho_n(t) = n^{-1} E \sum_{i=1}^n \delta(t - \lambda_i)$ for eigenvalues $\lambda_i$. We see in Thm. 4 that the expression for the expected number of critical points is given as a function of the eigenvalue density. On the other hand, the expected number of local minima is a function of the density of the largest eigenvalue of GOE matrices (cf. Thm. 9).

C  Proofs

Proof of Lem. 1. We can write the expression (3) in terms of the number of “active paths” to get rid of the nonlinearities $g'(.)$, i.e., all paths with non-zero weight from an input $X_i$ to the output $Y$. We then have

$$Y = \sum_{i=1}^n \sum_{\gamma \in \Gamma_i} X_i W_{\gamma}^i,$$  

(27)

where $\Gamma_i$ is the set of all active paths connecting the $i^{th}$ input neuron to the output $Y$. Let $W^{\gamma}_{i}$ be the weight along each path $\gamma$. Now define $\chi_{i_1 \ldots i_p}$ to be an indicator random variable that is 1 iff the path $i \rightarrow i_1 \rightarrow \ldots \rightarrow i_p \rightarrow Y$, i.e., the one that connects the $i^{th}$ neuron in the first hidden layer, $i^{th}$
neuron in the second hidden layer and so on, is active. With some abuse of notation, we will use the variable \( \gamma \) to denote both the path and the tuple \((i, i_1, i_2, \ldots, i_p)\). We now work towards removing the dependency of the input \( X \) in (27). First split the path into two parts, the first edge on layer 1 and the rest of the path as: \( \gamma_{i_1,i_2,\ldots,i_p} = \gamma_{i_1} \gamma_{i_1,\ldots,i_p} \) (conditional upon \( \gamma_{i_1} = 1 \), the sub-paths are independent). This gives

\[
Y = \sum_{i_1,i_2,\ldots,i_p=1}^{n} \gamma_{i_1} \gamma_{i_1} \gamma_{i_1} \ldots \gamma_{i_p} X_{i_1} W_{i_1,i_1} W_{i_1,\ldots,i_p}.
\]

This expression has correlations between different paths that are introduced by the sum over inputs \( X_i \); we approximate these correlations to obtain an expression that resembles the Hamiltonian in (4). Define the net coupling due to the sum over the inputs as

\[
Z_i := X_i \gamma_{i_1} W_{i_1}, \quad Z := \sum_{i=1}^{n} Z_i.
\]

Since \( X_i \in \{0,1\} \), we have \( \text{var}(X_i) \leq 1/4 \) from Popoviciu’s inequality (for a bounded random variable \( m \leq X \leq M \), the variance is bounded as \( \text{var} X \leq \frac{1}{4} (M-m)^2 \)), and \( W_{i_1,i_1+1} \) is positive or negative with equal probability \( d/2n \) while \( \gamma_{i_1} = 1 \) with probability \( d/n \) for a fixed \( i_1 \). Note that \( \text{E} Z_i = 0 \) while \( \text{E} Z_i^2 \leq d^2/(4n^2) \). We now use Bernstein’s inequality to see that:

\[
P \left( \sum_{i=1}^{n} Z_i > t \right) \leq \exp \left( -\frac{t^2/2}{\text{var}(Z)+t/3} \right)
\]

\[
\Rightarrow |Z| \leq n^{-1/2+\frac{1}{3p}} \leq n^{-1/5} \quad \text{for} \quad p > 5;
\]

with high probability for any small \( \varepsilon > 0 \) (we picked \( \varepsilon = 1/10 \) above). We have thus shown that the net coupling due to inputs in our model is bounded. Now note that \( \gamma_1 \gamma_2 \ldots \gamma_p \) is a Bernoulli random variable with

\[
P(\gamma_1 \ldots \gamma_p = 1) = \frac{dp}{n} \left( \frac{d}{2n} \right) \ldots \left( \frac{d}{2n} \right)^{p-1} \frac{1}{n} = \mathcal{O}(n^{-(p-1)/2}).
\]

Since the entries of weight matrices \( W^{(k)} \) are iid, the weights along each path \( \gamma \in \Gamma_{i} \), i.e., \( W_{i_1,\ldots,i_p} \) are independent and identically distributed, in particular, each \( W_{i_k,i_k+1} \) only depends upon \( i_k \). We therefore write them as a product of “spins” \( \sigma_1 \ldots \sigma_p \in \{-1,1\}^p \) to get

\[
Y \overset{\text{law}}{=} \sum_{i_1,i_2,\ldots,i_p=1}^{n} Z_{i_1,i_2,\ldots,i_p} \sigma_1 \ldots \sigma_p.
\]

We now approximate the Bernoulli random variable \( \gamma_1 \gamma_2 \ldots \gamma_p \) with a Gaussian of variance \( n^{1-p} \) and define the scaled output \( \hat{Y} = Y n^{1/5} \) to get

\[
\hat{Y} \overset{\text{law}}{=} \frac{1}{n^{(p-1)/2}} \sum_{i_1,i_2,\ldots,i_p=1}^{n} J_{i_1,\ldots,i_p} \sigma_1 \ldots \sigma_p,
\]

where \( J_{i_1,\ldots,i_p} \) is standard Gaussian. ■

**Proof of Lem. 3.** The zero-one loss \( L \) for \( Y' \sim \text{Ber}(q) \) can be written as

\[
L = \mathbb{E}_{Y'} \left| \hat{Y}(X') - Y' \right|
\]

\[
= q (1 - \hat{Y}) + (1 - q) \hat{Y}
\]

\[
= q + (1 - 2q) \hat{Y}.
\]

Now define \( H_{n,p} = (q - L)/(1 - 2q) \) to see that \(-H_{n,p} \) has the same distribution as \( \hat{Y} \). ■