Entropic alternatives to initialization

Daniele Musso*

Centro de Supercomputación de Galicia (CESGA),
s/n, Avenida de Vigo, 15705 , Santiago de Compostela, Spain

Abstract

Local entropic loss functions provide a versatile framework to define architecture-aware regularization procedures. Besides the possibility of being anisotropic in the synaptic space, the local entropic smoothening of the loss function can vary during training, thus yielding a tunable model complexity. A scoping protocol where the regularization is strong in the early-stage of the training and then fades progressively away constitutes an alternative to standard initialization procedures for deep convolutional neural networks, nonetheless, it has wider applicability. We analyze anisotropic, local entropic smoothenings in the language of statistical physics and information theory, providing insight into both their interpretation and workings. We comment some aspects related to the physics of renormalization and the spacetime structure of convolutional networks.

*daniele.musso@usc.es, mudaniele@yahoo.com

Contents

1 Introduction

Insight and methods coming from physics have ever since helped to improve our understanding and design of neural networks. The interdisciplinary potential of techniques borrowed from statistical physics and information theory, such as entropic regularizations and renormalization, has not yet exhausted its drive in machine learning.

Stochastic gradient descent (SGD) proved to be a particularly suited optimization algorithm to train deep neural networks. This is mainly due to the properties of its noise, which is related to the Hessian matrix of the loss function. Efficient escaping from sharp relative minima is an example of a useful effect directly descending from the characteristics of the SGD noise. Besides, it is argued that regularization effects due to noise bias the stochastic gradient descent algorithm towards encountering, both systematically and efficiently, solutions belonging to clusters characterized by a high value of the test accuracy. If not universal, this phenomenon is believed to be generic for networks working in a regime well below their critical capacity.

To the purpose of understanding the virtues of SGD algorithms from a theoretical viewpoint, and in order to define approaches able to enhance them, it has been recently proposed to modify the loss function by means of a local solution-counting term, a *local entropy*. Local entropy represents a refinement of standard entropy and defines a coarse-graining technique helpful in understanding and improving deep neural networks. In particular, it offers a tunable way of encouraging the training towards clusterized solutions through a smoothened version of the

original loss function. More precisely, the modified loss function allows us to perform a large-deviation analysis, biasing the statistical measure away from Gibbs typicality towards regions with a high density of high-accuracy weight configurations [1, 2].

The local entropy framework is attractive in many respects: the relation to statistical mechanics improves its interpretability; it admits time-dependent and anisotropic generalizations, making it a flexible framework allowing for adaptive strategies; it demonstrated a potential in image-classification experiments and constraint-satisfaction problems. Although being trained with a regularized loss function inspired by statistical mechanics, the deep networks used in practice are in general very far from being amenable to an analytical description. They, in fact, depart in many ways from the regimes where analytical approaches can be available.

Local entropy is associated to a position-dependent Helmholtz free energy defined by convoluting the Boltzmann weight with a specified kernel. The term *local* refers to the fact that the convolution kernel is significantly different from zero only over a compact region.[1] A natural example is provided by

$$e^{-\beta \mathcal{F}(\boldsymbol{W})} = \sqrt{\frac{\beta\gamma}{2\pi}} \int d\boldsymbol{W}' e^{-\beta\left[\mathcal{L}(\boldsymbol{W}') + \frac{\gamma}{2}\|\boldsymbol{W}-\boldsymbol{W}'\|_2^2\right]} , \tag{1}$$

where the original loss function $\mathcal{L}$ plays the role of the energy and β represents an inverse temperature. With $\boldsymbol{W}$ we denoted the position in the synaptic space (*i.e.* a vector collecting the weights of the network, thus representing its state), $\|\|_2$ is the Euclid-Frobenius norm and γ is inversely related to the width of the Gaussian kernel

$$K(\boldsymbol{W}, \boldsymbol{W}') = e^{-\beta\frac{\gamma}{2}\|\boldsymbol{W}-\boldsymbol{W}'\|_2^2} . \tag{2}$$

The idea is to use $\mathcal{F}(\boldsymbol{W})$ as a regularized version of the original loss, where the smoothening scale is controlled by $\gamma^{-\frac{1}{2}}$.

Even before considering more generic kernels K, (1) can be extended naturally in two ways: on the one hand, one can consider anisotropic Gaussian kernels where γ takes a different value for different subspaces in the synaptic space. This corresponds to weighting differently the distance between two points depending on the direction of their separation and has been dubbed *partial local entropy* [3]. On the other hand, one can consider time-dependent choices where γ follows either a pre-defined schedule or an adaptive protocol.

The present work is concerned with such generalizations, devoting particular attention to the time-dependent extension of (1), which -as we will show- provides a useful alternative to sophisticated initialization procedures in image-classification tasks performed with deep convolutional networks. Furthermore, a varying γ admits practically useful insight coming from complexity theory and the physics of renormalization.

2 Architecture-aware entropic regularization

The main point of considering a different γ for different directions in the synaptic space, namely considering an anisotropic local entropy, consists in treating distinct weights in a different manner. Since deep neural networks are by construction hierarchical systems, where the nature of

[1] Spatial locality in a high-dimensional space like the synaptic space can be a somewhat misguiding concept. Indeed, recall that -in a high-dimensional space- the volume of a sphere is sharply concentrated close to its surface. A random sampling of such high-dimensional sphere, when uniform in volume, would thereby concentrate in the near-surface region.

the hierarchy is connected to combinatorial complexity, the anisotropic extension of (1) is theoretically natural. Said otherwise, it seems in general not adequate to treat all the weights on the same footing as far as regularization is concerned.

Assigning different values of γ to different weight subspaces (*e.g.* a different γ for each layer) corresponds to defining a regularization strategy which is adapted to the network architecture. It is therefore fair to expect that a suitable, anisotropically tuned, γ can permit a better exploitation of the biases intrinsically hard-coded in the architecture itself.

These comments appear particularly suited to deep networks, where the depth has a crucial and transparent role. Nonetheless, the idea can be generalized to other contexts. In general, it amounts to having a tunable neural sensitivity and -as such- it can be connected to studies on neural plasticity. More theoretically, it is interesting to use the convolutional kernels introduced through (1) as a probe to explore (and modify) the local capacity and sensitivity of the network.[2]

The analysis of [3] considered anisotropic entropic regularization where the anisotropy in the synaptic space respects the layer structure of the network. That is, all the neurons belonging to a same layer are smoothened over in the same way. In this sense, the entropic regularization can be architecture-aware. As a special case, one can consider entropic regularization on a single layer at a time. The numerical experiments described in [3] hinted to the possibly generic fact that single-layer regularization is more effective than multi-layer regularization, on the one side, and increasingly more effective when applied to increasingly deeper layers, on the other.[3] We corroborate this observations with a new series of experiments performed on MNIST with a 5-layer fully-connected cylindrical network where all the layers, except the output layer, have 784 neurons. For the training we have considered momentum $\mu = 0.9$, constant learning rate $\eta = 10^{-4}$, constant batch-size of 256 images, ReLU activations and no weight-decay regularization. The results are reported in Figure 1, which calls for some comments. First, the asymptotic test-accuracy appear to define a "discrete spectrum". This reinforces the idea that the SGD training encounters solutions belonging to few clusters, representing rare deviations from typicality, yet systematically found. Considering the same regularization intensity on progressively deeper layers enhances the performance, indicating that the entropic regularization is more effective when applied to the synapses corresponding to more complex features. This seems to agree with the intuitive idea that the same level of noise is more harmful when affecting a deep rather than a shallow layer.

3 Information theoretic interpretation of local entropies

It is useful to study the connections of local entropy to information theory. This allows us to appreciate how the modified loss function (*i.e.* the local free energy) is associated to an entropy encoding the similarity of the convolutional kernel and the modified Boltzmann weight. More precisely, the local entropy arises from a relative entropy.

Let us first introduce a modified (local) partition function as

$$\tilde{Z}(\boldsymbol{W}) = \int d\boldsymbol{W}' e^{-\beta \mathcal{L}(\boldsymbol{W}')} K(\boldsymbol{W}, \boldsymbol{W}') \ . \tag{3}$$

[2] More comments on the relation between the kernel size and measures of network complexity are given in Section 6.

[3] The situation can actually be more complicated than just stated, and -for instance- it may depend on the radius of the vicinity over which one smoothens the loss function; we refer to [3] for a more detailed discussion. The last output layer, having as many neurons as the classes of the task, is actually excluded from the argument. Thus, the "deepest" layer is the second-last from the input.

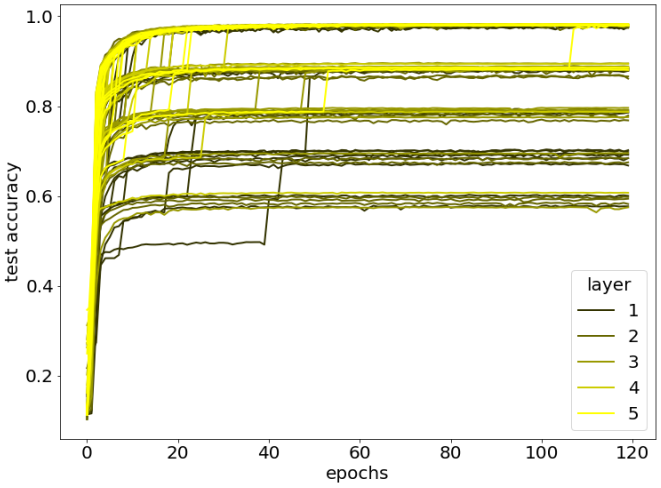

Figure 1: Single-layer entropic regularization is more effective when applied to progressively deeper layers.

For the moment being, we leave the convolutional kernel K generic, yet we assume that it satisfies

$$K(\boldsymbol{W}, \boldsymbol{W}') \geq 0 \ , \qquad \text{for all} \qquad \boldsymbol{W} \ , \ \boldsymbol{W}' \ , \tag{4}$$

and

$$\int d\boldsymbol{W}' \, K(\boldsymbol{W}', \boldsymbol{W}) = \int d\boldsymbol{W}' \, K(\boldsymbol{W}, \boldsymbol{W}') = 1 \ , \qquad \text{for all} \qquad \boldsymbol{W} \ . \tag{5}$$

This corresponds to a normalization property[4] thanks to which we have

$$\int d\boldsymbol{W} \tilde{Z}(\boldsymbol{W}) = \int d\boldsymbol{W}' e^{-\beta \mathcal{L}(\boldsymbol{W}')} \int d\boldsymbol{W} K(\boldsymbol{W}, \boldsymbol{W}') = \int d\boldsymbol{W}' e^{-\beta \mathcal{L}(\boldsymbol{W}')} = Z \ , \tag{6}$$

where Z is the standard partition function. Thus, $\tilde{Z}(\boldsymbol{W})$ represents the local contribution to Z.[5]

As customary in statistical mechanics, the free energy is derived from the partition function through

$$\tilde{F}(\boldsymbol{W}) = -kT \ \ln \tilde{Z}(\boldsymbol{W}) \ , \tag{7}$$

which provides a local generalization of the standard derivation. We have introduced a Boltzmann constant k and the temperature T (such that $\beta = \frac{1}{KT}$) to maintain the connection to statistical physics explicit, yet the experiments will be be performed taking $\beta = 1$.

The partition function (3) allows us to define the following probability distribution

$$\tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') = \frac{e^{-\beta \mathcal{L}(\boldsymbol{W}')}}{\tilde{Z}(\boldsymbol{W})} K(\boldsymbol{W}, \boldsymbol{W}') \ . \tag{8}$$

[4]For the specific case of (1) the normalization was encoded in the $\sqrt{\frac{\beta\gamma}{2\pi}}$ pre-factor.

[5]A tilde is adopted throughout the paper to represent the local generalization of the tilded quantity.

Note that the probability distribution is to be thought of as $\tilde{\rho}_{\boldsymbol{W}}(\boldsymbol{W}')$, that is, a probability distribution over the synaptic space spanned by $\boldsymbol{W}'$ where $\boldsymbol{W}$ plays the role of a parameter (namely, the "center" about which we define local quantities).

From the definition of the partition function (3), we have the correct normalization of $\tilde{\rho}$, namely

$$\int d\boldsymbol{W}' \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') = 1 \ . \tag{9}$$

A re-arrangement of the terms in (8) gives us

$$\beta \mathcal{L}(\boldsymbol{W}') = -\ln \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') - \ln \tilde{Z}(\boldsymbol{W}) + \ln K(\boldsymbol{W}, \boldsymbol{W}') \ , \tag{10}$$

which will be useful shortly.

Now, we follow the standard steps to define the entropy, yet we start from the local free energy (7), namely

$$\begin{aligned}
\tilde{S}(\boldsymbol{W}) &= -\frac{\partial \tilde{F}(\boldsymbol{W})}{\partial T} \\
&= k \left[\ln \tilde{Z}(\boldsymbol{W}) + \frac{\beta}{\tilde{Z}(\boldsymbol{W})} \int d\boldsymbol{W}' \mathcal{L}(\boldsymbol{W}') e^{-\beta \mathcal{L}(\boldsymbol{W}')} K(\boldsymbol{W}, \boldsymbol{W}') \right] \\
&= k \left[\ln \tilde{Z}(\boldsymbol{W}) - \int d\boldsymbol{W}' \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') \ln \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') \right. \\
&\qquad \left. - \int d\boldsymbol{W}' \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') \ln \tilde{Z}(\boldsymbol{W}) + \int d\boldsymbol{W}' \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') \ln K(\boldsymbol{W}, \boldsymbol{W}') \right] \\
&= -k \int d\boldsymbol{W}' \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') \left[\ln \tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') - \ln K(\boldsymbol{W}, \boldsymbol{W}') \right] \\
&= -k \, D_{\mathrm{KL}} \left[\tilde{\rho}(\boldsymbol{W}, \boldsymbol{W}') \middle|\middle| K(\boldsymbol{W}, \boldsymbol{W}') \right] \ .
\end{aligned} \tag{11}$$

Equation (11) shows that the local entropy $\tilde{S}(\boldsymbol{W})$ corresponds to the relative entropy among the probability distribution (8) and the kernel K [4, 5]. To gain intuition, equation (8) implies that a flat loss $\mathcal{L}(\boldsymbol{W})$ maximizes $\tilde{S}(\boldsymbol{W})$. Note that, being the kernel K present in the definition of the distribution $\tilde{\rho}$, the relative entropy (11) encodes mainly an intrinsic property of $\mathcal{L}(\boldsymbol{W})$ rather than a property induced by the shape of the kernel. Yet, the kernel is determining the region in synaptic space upon which the entropic comparison is made.[6]

4 Entropic regularization and tunable complexity

The number of data points introduces a natural resolution scale into the synaptic space [6]. To rephrase, the complexity of the dataset translates into the complexity of the landscape of the loss function built upon the dataset itself. In this perspective, a regularization techniques which smoothens the loss function reduces the complexity of the landscape. This can be precisely stated in terms of the Fisher information matrix [6]. Indeed, a smoothened loss will have –by construction– a softer dependence on the weights, leading therefore to smaller eigenvalues of the Fisher information matrix.[7]

[6] Abusing the language to the sake of conveying an intuitive idea, one could say that local entropy is an entropy with a *receptive field*, this latter being determined by the support of the considered convolutional kernel K.

[7] We will further comment this point in relation to local scale invariance in Subsection 6.1. For the definition of the Fisher information matrix and its relevance to the present discussion we refer to [5] and [6], respectively.

The possibility of tuning by hand the complexity of the synaptic space is attractive both on a theoretical and on a practical level. Roughly, it corresponds to having a tunable sensitivity for the synapses, which can be exploited to define improved training protocols. In practice, this amounts to consider an entropic regularization like (1) where the regularization parameter(s) γ evolves in time during training. Its evolution can either be ruled by a pre-determined schedule or be adaptive. The former possibility corresponds to a planned *scoping* of the entropic regularization, the latter introduces an extra intrinsic dynamical ingredient.

The fact that scoping the local entropic regularization can be a good idea is directly suggested by experiments where no scoping is considered. In fact, when considering sufficiently complicated image-classification tasks like CIFAR10 or STL10, it appears that the entropic regularization provides an advantage in training performance only when combined with an early-stopping protocol [3]. In other words, the local entropic regularization seems helpful, but only in an early stage of the training process.

Such phenomenon can be understood as follows. The early stage of a stochastic gradient descent is typically noisier than later stages. Thereby, filtering away some noise in the initial phase produces a stabler and more effective training signal.[8] Nevertheless, at later training stages, the information filtered away by a smoothening process can be useful to further optimize the network. This corresponds to the fact that a coarse-grained loss would perform asymptotically in a sub-optimal fashion.

A further, more theoretical reason in favor of switching off the entropic smoothening along the training connects to convergence. If the regularization is switched off completely starting from some (possibly predetermined) training step, then we can directly apply the convergence results of the standard stochastic gradient descent algorithm to the overall training.

Another related way to interpret the effect of an entropic smoothening is as a device enforcing an exploration/exploitation (or robustness-sensitivity) trade off. Progressively switching off the entropic smoothening appears to be desirable, amounting to favoring exploration in an early phase of the training, while enhancing the exploitation later on. Indeed, we want to be more robust initially and then learn finer details in a subsequent training phase, which is implemented through a *"search-then-converge"* schedule [7] for the entropic regularization.

It is interesting to draw a comparison with renormalization theory in statistical mechanics, which requires a small detour. Renormalization in statistical physics can be described as a (systematic) procedure to filter away the information about the microscopic dynamics of a physical system in order to retain only the *relevant* dynamics determining its low-energy or overall behavior.[9] As an extreme case of renormalization, consider for instance the thermodynamic description of a gas where a small set of thermodynamic parameters (the temperature, the pressure and so on) accounts for the overall description of the macroscopic behavior of a microscopically very complicated system.

In a machine learning task, to some extent, one proceeds as in renormalization: one is interested in filtering away (irrelevant) information about the details of the dataset to keep just the relevant information needed for the task. Thus, seemingly, a training process should be interpretable as a dissipative flow along which irrelevant information is progressively forgot.

This intuitive picture is not sufficient to account for the training of a neural network. To understand this it is enough to note that, unlike the physical system (stick to the example of a gas), the neural network does not contain the microscopic information to begin with. In other words, the training is not simply a filtering operation, rather is it a process in which information is

[8] This can be put in analogy to the effect of momentum, we discuss this comparison in Subsection 6.4. We also refer to the discussions on the effects of momentum contained in [7, 8].

[9] See Appendix A for related comments.

acquired from the dataset and -generically- filtered at the same time.[10] In this sense, the scoping of the entropic smoothening corresponds to organizing the learning priority, coarse-grained and robust features first, finer details later on.

5 Alternative to initialization

Since the entropic smoothening can provide tunable parameters which control the synaptic sensitivity, it can be exploited to pursue an alternative solution to the so-called *initialization problem* in deep networks. Complicated neural networks, especially those that -due to their depth- entail strong hierarchies among different weights, can need suitable initialization procedures in order to be trained. This is the case for deep convolutional neural networks. Although the initialization problem has been studied in detail, and suitable as well as easy initialization procedures have been devised [12, 13], it is interesting to consider alternatives. In particular, one would like to seek for methods which can be applicable in general, independently of the specific characteristics of the architecture of the underlying neural network. In this context, the entropic smoothening offers a viable and versatile tool. By considering an aggressive entropic regularization at the beginning of the training, one induces a strong insensitivity to the initial state.

The idea of an initially insensitive neural network, which is progressively made more sensitive during training matches with the arguments described in Section 4. Here we stress that such scoping of the entropic smoothening can be pushed to the extent that it makes an initialization procedure superfluous. To substantiate this proposal we tested it in two different circumstances, both referring to image-classification tasks. We describe the experimental details and results in two separate subsections, Subsection 5.1 for experiments performed on the MNIST dataset and Subsection 5.2 for experiments performed on STL10.

It is useful to stress that here we are considering an entropic regularization on the weights of convolutional layers. This has been argued to worsen the performance of the neural network [3].[11] However, here we consider a regularization which is active only at an early stage of the training and which fades away completely at later stages.

Another technical observation, which applies to all the experiments described below, is that the entropic regularization has been enforced by taking a single extra weight configuration for each training step. This corresponded to considering a kernel K given by the characteristic function of a hypercube and approximating the convolution integral (3) by means of a (minimal) empirical sampling. The extra configuration $\boldsymbol{W'}$ is sampled uniformly in a hypercubic vicinity of the original configuration $\boldsymbol{W}$. More precisely, the extra sampling point $\boldsymbol{W'}$ is generated as a perturbation of the unperturbed configuration $\boldsymbol{W}$ according to

$$\boldsymbol{W'} = \boldsymbol{W} + \Delta \boldsymbol{W} \ , \tag{12}$$

where $\Delta \boldsymbol{W}$ is a vector whose components ΔW_i are uniformly distributed in the interval

$$[-R_i, R_i] \ . \tag{13}$$

The parameter R_i sets the size along the i-th direction of the hypercube.[12] Note that R_i controls the size of the smoothening vicinity, a role that in (1) corresponded to $\gamma^{\frac{1}{2}}$.

layer	input channels[13]	output channels	kernel size
conv	1	1	3
conv	1	1	3
conv	1	1	3
conv	1	1	3
fully conn.	$20 \cdot 20$	10	

Table 1: Deep convolutional architecture adopted for image classification on MNIST.

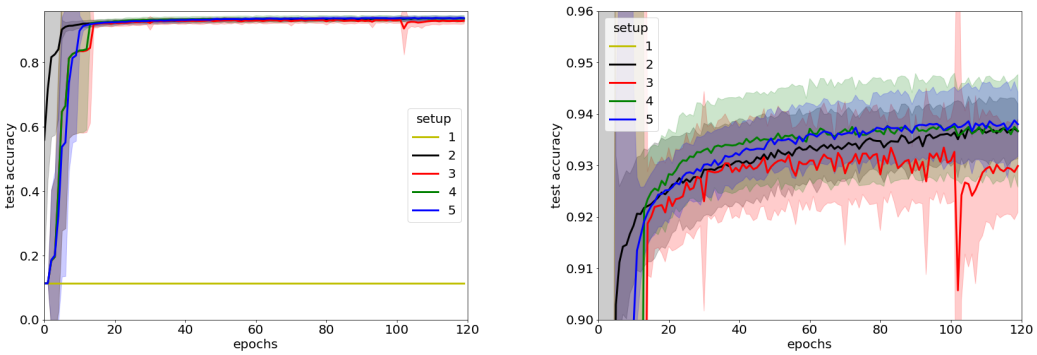

Figure 2: The two plots show the same training experiments on MNIST, but the one on the right zooms into the high-accuracy region. The numbering in the legend corresponds to the list of protocols described in the main text.

5.1 MNIST

The convolutional architecture employed in the series of experiments on MNIST is detailed in Table 1. The entropic smoothening has been applied only on the weights belonging to the convolutional layers, leaving those of the fully-connected head un-regularized. No weight decay or other sources of regularization on the weights have been considered. The learning parameter has been kept always constant in time $\eta = 0.001$, the same is true for the mini-batch size $C = 256$ and the momentum $\mu = 0.9$. The neural network has been trained for 120 epochs and we have considered five distinct protocols in relation to initialization and the entropic regularization schedule,

1. Random initialization according to a normal distribution with zero mean and $\sigma = 0.01$. No entropic regularization.

2. Kaiming initialization [13]. No entropic regularization.

3. Random initialization according to a normal distribution with zero mean and $\sigma = 0.01$. Constant, anisotropic entropic regularization according to $R_i = (i-1)\,\sigma$ with the index i

[10] These comments are close to the information bottle-neck analysis of neural networks, see [9–11].
[11] A similar observation applies to dropout regularization of convolutional layers which generically affects negatively the overall performance [14].
[12] See [3] for details.

layer	input channels	output channels	kernel size
conv	3	8	3
conv	8	8	3
max pool			2
conv	8	16	3
conv	16	16	3
conv	16	16	3
max pool			2
fully conn.	$16 \cdot 20 \cdot 20$	$16 \cdot 20 \cdot 20$	
fully conn.	$16 \cdot 20 \cdot 20$	$16 \cdot 20 \cdot 20$	
fully conn.	$16 \cdot 20 \cdot 20$	10	

Table 2: Deep convolutional architecture adopted for image classification on STL10.

counting the layers, $i = 1, ..., 4$.[14]

4. Random initialization according to a normal distribution with zero mean and $\sigma = 0.01$. Scheduled, anisotropic entropic regularization starting with $R_i = (i-1)\,\sigma$, reduced by a factor $\frac{1}{3}$ for each ten training epochs (the scheduling corresponds to an exponential decay).

5. Random initialization according to a normal distribution with zero mean and $\sigma = 0.01$. Scheduled, anisotropic entropic regularization with $R_i(t) = \frac{i-1}{\sqrt{t}}\,\sigma$ where t is a discrete time variable counting the number of training epochs elapsed.

The results are summarized in Figure 2. The random initialization with no entropic regularization (protocol 1) could not be trained. Random initialization with a constant entropic regularization (protocol 3) proved to be trainable but led to sub-optimal results. The remaining three protocols proved to be optimal and -essentially- equivalent. To recapitulate, a training starting from random initialization, when performed according to a properly scheduled entropic regularization, led to equivalent results as a non-regularized case initialized with the Kaimimg method.

5.2 STL10

We still consider a series of experiments similar to those described in Subsection 5.1, here performed on the significantly more demanding image-classification task defined by the STL10 dataset. To this purpose, we adopt the architecture described in Table 2. The network is one layer deeper than that used on MNIST, but -apart from this difference- we consider the same experimental setups as described in the list in Subsection 5.1. Also regarding the training hyperparameters, we consider the same values as described there, namely, $\eta = 0.001$, $C = 256$ and $\mu = 0.9$.

As it already happened for MNIST, also on STL10 the random initialization with no entropic regularization (protocol 1) corresponds to a setup where the network does not train. The case where the entropic regularization is kept constant throughout the training with random initialization (protocol 3) is trainable but -again- it leads to suboptimal results. Actually, adopting protocol 3, a sufficiently long training can spoil completely the results obtained at an earlier stage of the training. Protocols 2 and 5 proved to be practically equivalent, as far as the asymptotic

[14] In (13) the index i ran over the weights, here we overload the notation because we are considering that R_i is equal for all the weights belonging to the same layer.

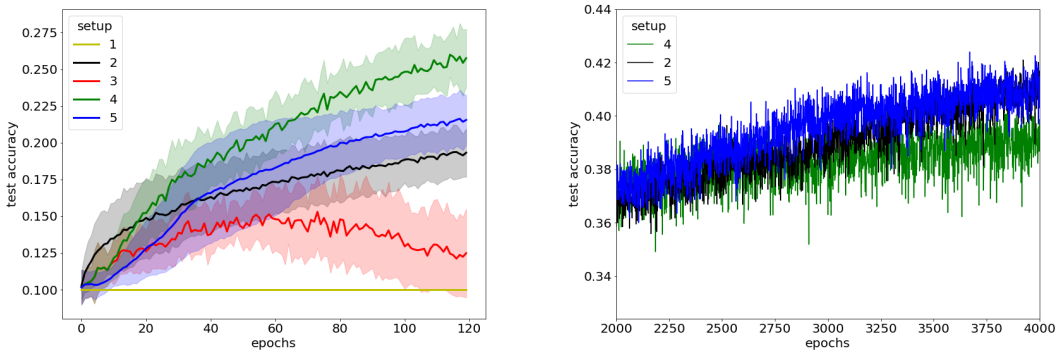

Figure 3: Training on STL10 according to the protocols detailed in Subsection 5.1 (the same as those adopted on MNIST). Left: early training phase. Right: large-time behavior for protocols 2, 4 and 5.

test accuracy is concerned, with protocol 4 reaching a slightly sub-optimal result. Nonetheless, the training dynamics is quite different among the three protocols. A scheduled entropic regularization, especially when switched off exponentially, proved to be the best achieving protocol in a wide portion of the early training.

The experiments on STL10 features a richer structure with an interest on its own (especially in relation to early-stopping policies). They however corroborate the conclusions already reached on the experiments on MNIST: a progressively fading entropic regularization can provide a valid alternative to Kaiming initialization.

6 Discussion

6.1 Local scale invariance

We adopt the Rectified Linear Unit (ReLU) activation function for all the neurons in the network, which is a scale covariant function, namely

$$\sigma_{\mathrm{ReLU}}(\alpha\, x) = \alpha\, \sigma_{\mathrm{ReLU}}(x)\ , \tag{14}$$

where α is a generic real constant. The outputs z_j of the network are normalized by means of a *softmax* function,

$$\hat{p}_j = \frac{e^{-z_j}}{\sum_k e^{-z_k}}\ , \tag{15}$$

so that $\{\hat{p}_j\}$ can be interpreted as an empirical probability distribution over classes, in the Bayesian sense of encoding a degree of uncertainty.

The two characteristics expressed in (14) and (15) make the network prediction independent from a scaling of all the weights of a layer by the same constant factor. We refer to this property as *local scale invariance*, where the "local" attribute refers here to depth, namely we can have a different scaling factor α_i for each layer i. Clearly, local scale invariance is stronger than global scale invariance, this latter corresponding to $\alpha_i = \alpha$ for all i.

If the weights of a layer are distributed with a variance v_i, scaling them by a factor α transforms the variance to $\alpha^2 v_i$. From this observation it emerges that the initialization procedures, which tune the variance of the initial weight distribution, set a local scale (*i.e.* a scale for each layer), meaning that the initialization does not commute with a local scaling transformation. If we think to the cost function as an energy[15] and to the weight configuration as the state of the system, then we have that local scale invariance is preserved by the energy function (*i.e.* the energy is invariant under local scale transformations) but broken by the state of the network. This corresponds to what in physics is referred to as *spontaneous symmetry breaking*.

The former statement applies to the unregularized loss function $\mathcal{L}(\boldsymbol{W})$, before considering the entropic regularization (1). In fact, the regularizing term in (1) is not invariant under local rescalings of the weights, thus neither it is so the *local free loss* $\mathcal{F}(\boldsymbol{W}')$.[16] This situation is referred to as *explicit breaking* of the local scaling symmetry. Indeed, one can think to

$$\mathcal{L}_\gamma(\boldsymbol{W}, \boldsymbol{W}') = \mathcal{L}(\boldsymbol{W}) + \frac{\gamma}{2}\|\boldsymbol{W} - \boldsymbol{W}'\|_2^2 \,, \tag{16}$$

appearing at the exponent in the integrand of (1), as a modified energy function, where γ is a source of explicit symmetry breaking. In other words, γ introduces explicitly a scale into the formerly scale-invariant problem. In the case in which we consider different γ's for different directions in the synaptic space, we introduce more than one explicit scale into the problem.

It is relevant to note that these statements about local scaling symmetry apply also when considering the training dynamics. Specifically, if the loss is invariant under local scalings, then the training step (*i.e.* the gradient descent step) commutes with a local rescaling. This means that one can perform the rescaling and the training step in either order and reach the same final state. This is due to the linearity of the gradient descent algorithm and would cease to apply when considering higher-order descent algorithms.

Explicit breaking of the local scale invariance, either implemented by means of a local-entropic loss function or by considering activation functions which are not scale covariant, can be contrasted with the so-called *natural gradients* approach [15,16]. This latter employs a normalization technique, based on approximated information-theoretic arguments, to avoid the scaling ambiguity. More recently, normalization techniques aimed at removing the local scaling ambiguity have been studied in [17, 18].

Although the local scale invariance is in many respects similar to a redundancy which we have to project away[17], the considerations about its breaking, either spontaneous or explicit, might have a relevant role. Specifically, one could consider how the interplay between the spontaneous scale introduced by initialization and the explicit scale introduced by local entropic regularization affect the network training and eventual performance.

There are at least four further observations which connect to local scale invariance:

- Scale invariance emerges in common but very specific setups, for instance, in networks characterized by the exclusive adoption of ReLU activations. These systems can thus be regarded as a fine-tuned family within the wider set of possible networks that are not scale invariant. Apart from their practical interest [19], scale-covariant activations are simpler to study. As such, they can constitute a good starting point in view of studying how the scales introduced by non-covariant activations may affect the behavior of the networks.[18]

[15] To the purposes of the present discussion, one can interpret the cost function as the static potential of a would-be Hamiltonian system.

[16] To avoid confusion, we remind ourselves that the adjective *local* in *local free loss* refers to locality in the synaptic space.

[17] In theoretical physics, one would speak of a gauge invariance which needs to be fixed.

[18] We leave a systematic experimental study of these aspects to the future, especially because the theoretical

- The regularized loss function $\mathcal{F}(\boldsymbol{W})$ introduced in (1) is defined in terms of a convolution integral over the synaptic space. Although it introduces into the problem an explicit scale through γ, this does not *lift* the flat directions of the original loss associated to local scale invariance. Equivalently, both the original loss $\mathcal{L}(\boldsymbol{W})$ and the regularized one, $\mathcal{F}(\boldsymbol{W})$, have a Fisher matrix with some zero eigenvalues associated to the local scaling freedom.

- As already commented in Section 4, complexity theory and, specifically, the concept of *effective capacity* [6] suggest that also the dataset can induce the notion of a scale into the synaptic space. This corresponds intuitively to the idea that a richer dataset defines a more detailed loss.[19]

- Individuating the physical scales and the hierarchies among them is at the basis of possible effective descriptions of neural networks [20–22]. Scale transformations generate the renormalization group whose fixed points are associated to criticality, a condition, or regime, which could radically simplify the analysis and the training of the networks [23]. Entropic probes have been considered recently in this context, see [24].

6.2 Related frameworks

In practice, the computation of the convolution integral (1) is not convenient as it would entail a significant extra cost. Rather, one considers an empirical proxy for such an integral obtained by sampling a discrete set of points according to a suitable sampling criterion. A similar approach is considered in *entropic least action learning*, where a number of real copies of the original system are trained concomitantly and coupled to one another by means of an attractive interaction, whose intensity is typically increased along training [25, 26]. Such *elastic regularization* implements a soft version of a distance constraint between the machines working in parallel [27, 28].

Another framework exploiting parallelism to enforce an entropic bias and, more generically, to enhance the exploratory character of an algorithm is quantum annealing (see for instance [29]). In quantum annealing the support of the wave-function corresponds to the explored region. In such a framework, an external control enhancing the focusing of the wave-function along the training would -at least intuitively- parallel the increasing binding interaction among parallel machines in least-action learning.

6.3 Minimal extra cost

As already mentioned, local entropic regularizations can be approximated by suitable sampling techniques. The sampling, although being in general cheaper than the actual computation of the integral, amounts to extra evaluations of the loss function and computation of its gradient. Clearly, this correspond to an increased computational cost at each training step. Two relevant comments in this respect are the following.

- Numerical experiments show that the cost can be minimal, yet still leading to useful effects. In particular, already adding just one extra sampling point in the vicinity of the original

picture is likely more involved (and probably more interesting, too). Nonetheless, we give some further comments on this in Subsection 6.6.

[19] It is relevant to observe that fixing the local scale invariance by means of normalizing the weights of each layer defines a sphere for each subspace (of the synaptic space) corresponding to a layer. In other words, if $\boldsymbol{W}_i$ is a vector whose components represent the weights of the i-th layer, its normalization means that $\boldsymbol{W}_i$ spans a spherical surface. Such surface is compact, and compactness is crucial to argue that a finite dataset can induce a physical scale in the synaptic space.

point yields the positive effects of local entropic regularization.[20]

- The extra samplings do not require re-loading the image mini-batch, which is in general a costly operation. This represents the main difference between the approach described here from that usually adopted in entropic least action learning (see Subsection 6.2) where each machine working in parallel is provided with a different mini-batch of training samples.

6.4 Gradient de-noising

A local entropy regularization extracts more information at each training step by sampling the point and its vicinity. As such, it has a *de-noising* effect on the computed gradient. Momentum, too, has a denoising effect on the training gradient.

Momentum integrates information coming from previous training steps. It does not entail an increase of training cost, but only in memory resources (which should be generically cheap). Conversely, partial local entropy exploits spatially and temporally local information. It requires an additional sampling cost (see Subsection 6.3). Momentum is "conservative", exploiting information coming from the past and resisting to update it by means of an inertial update rule; partial local entropy, instead, exploits as much as possible the current circumstance independently of how one has reached it. The two things might not be as independent as they seem at first: the current weight configuration is conditioned by previous training so, even if we decide to forget the gradient information collected in the previous steps, there is some information correlated to it encoded in the current state. Yet, this would be very difficult to disentangle explicitly.

6.5 Size and shape of the solution clusters

Stochastic gradient descent and its regularized versions typically encounter solutions belonging to clusters of configurations leading to similar test accuracy. The size of the solution cluster, that is, the Gardner volume, can be assessed with entropic probes, either analytically (when treatable) or numerically [19, 30]. In a setup where the entropic regularization is non-trivial at the end of the training, the final γ encodes a bias on the size of the encountered cluster of solutions. Therefore, a suitably anisotropic γ can be used as a probe of the shape of the solution cluster [3]. In this context, we should recall that -if working with a scale invariant network- the concept of shape is meaningful only after a suitable normalization scheme has been adopted. For instance, normalizing all the layers to a fixed and pre-determined value. This latter observation represents a refinement on the critical observations about the concept of "wide valleys" discussed in [17].

6.6 Activation functions which are not scale covariant

The activation functions can have a relevant impact on the properties of the loss landscape. We have already seen an explicit example of this when discussing scale invariance in Subsection 6.1 and, specifically, when observing that the scale covariance of ReLU activations produces some exactly flat directions in the loss landscape. As shown both analytically and numerically in [19], the robustness properties of the solution clusters may depend on the choice of activation functions.

[20] A related observation was already made in [3], where experiments performed with a bi-layer fully-connected network on the Fashion-MNIST image-classification task yield very similar results irrespective of the fact that the partial entropic regularization relied on either 5 or 9 sampling points. Note that these numbers are very small with respect to the dimensionality of the problem (the number of weights). For a connection between the number of sampling points and the Parisi parameter m see [19].

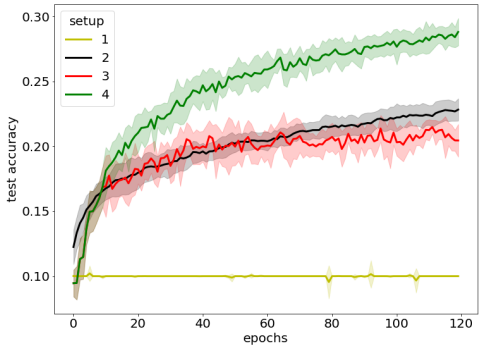

Figure 4: Test accuracy during training for a convolutional neural network specified in Table 2, but with tanh activations instead of ReLU. The task, image classification on STL10, and the hyperparameters are the same as those described in Subsection 5.2.

Considering a non-scale covariant activation function, namely tanh, lifts the flat directions associated to the local scale transformations discussed in Subsection 6.1. In particular, the ratio between the intrinsic scale dictated by the tanh activation (related to the steepness of the transition between the two asymptotic saturating behaviors) and the scale (or scales, in the anisotropic case) enforced by γ could produce some observable effects in the training dynamics. Yet, some preliminary experiments -performed on STL10 with an identical setup as the one described in Subsection 5.2 where ReLUs have been substituted with tanh activations- show a training behavior which is qualitatively analogous. Compare Figures 4 and 3.

6.7 Dynamical landscape

Suppose that the training dynamics has some time dependence on top of the scoping of the γ parameters or a scheduling of the learning rate. Namely, some time dependence which is related to the task evolution in time. An explicit example of this can be provided by a dataset which changes over time. If the task evolution is rapid enough, we can think that the training dynamics finds itself constantly in a situation similar to being in an early training stage. As argued in Section 5, this is the circumstance in which an entropic regularization can provide an advantage, for convolutional weights too. Thus, we can think that $\gamma(t)$ should depend on time in response to the variations of the dataset. Roughly, we need a higher γ when the dataset variations are stronger, while we need a decaying γ when the dataset is static or quasi-static. To rephrase, $\gamma(t)$ seems to need to be adaptive and related to the time-dependent properties of the task.[21] In physical terms, $\gamma(t)$ should adequately respond to the external driving. Entropic regularization could therefore be interesting in the context of the problems related to catastrophic forgetting [31].

[21] In this sense, the initial condition of a training with a constant dataset can be interpreted as an abrupt change, or a quench, separating the time before training from the training time. As such, the early training profits from a strong entropic regularization as a response to the initial conditions.

6.8 Final remarks

We showed that an anisotropic and suitably scheduled entropic regularization can provide a tunable alternative to initialization procedures for deep neural networks. Moreover, the entropic protocols can enhance the test-accuracy, especially in early training phases. The training dynamics featured by the entropic protocols appears to be very rich and sensitive to both the architecture and hyper-parameters, on the one side, and to the characteristics of the task, on the other. As such, a systematic characterization of the effects of anisotropic and scheduled entropic regularization is a very complex and wide task. We provided here a first exploratory round of experiments to stress the potential and the theoretical interest. An in-depth experimental study constitute however a promising route for future investigations, which could be performed with either an exploratory or an exploitative attitude, that is to say, it can be pursued with a tunable inclination towards practical applications.

7 Acknowledgements

A special acknowledgment goes to Manuel Fernández Delgado and Giorgio Musso for interchanges and feedback on the draft.

I would like also to thank Amparo Alonso Betanzos, Antonio Amariti, Carlo Baldassi, Brais Cancela Barizo, Xabier Cid Vidal, Aldo Cotrone, Thomas Dent, Carlos Eiras Franco, Andrés Gómez Tato, Carlos Hoyos, Alessandro Ingrosso, Estelle Maeva Inack, Esteban Requeijo Gonzalez, Lorenzo Rosasco, Silvia Villa and Riccardo Zecchina for stimulating and interesting conversations.

A Entropic smoothening as a filtering process

Consider the definition of the local partition function (3) as a *filtering* procedure applied to the Boltzmann weight $e^{-\beta\mathcal{L}(\boldsymbol{W})}$ and expressed mathematically through the convolution with the integration kernel K.

To gain intuition about this, it is useful to think of two extreme cases. First, take the trivial kernel

$$K(\boldsymbol{W},\boldsymbol{W}') = 1 \ . \tag{17}$$

In this case the filtering due to (3) reduces to the simple integral of the Boltzmann weight over the configuration space. This is the standard definition of the partition function in statistical mechanics. In a learning perspective, it corresponds to having filtered away as much information as possible, just retaining the thermodynamic information actually encoded in Z.

An opposite circumstance occurs when the integration kernel takes the form of a Dirac delta,

$$K(\boldsymbol{W},\boldsymbol{W}') = \delta(\boldsymbol{W},\boldsymbol{W}') \ . \tag{18}$$

In such a case, the convolution (3) simply returns the original Boltzmann weight, the delta corresponding to an identity operator. Thus, no information has been integrated away by the convolution.

The cases with generic kernels K, as well as the specific examples considered in the main text (like that of a Gaussian K), fall intuitively between the two extreme cases just described. In other terms, partial local entropy can be interpreted as a procedure which refines the thermodynamic description, but still filters some microscopic information away. Specifically, local thermodynamic functionals defined by means of a convolution with a kernel with compact support (or, at least,

a kernel which is significantly different from zero on a compact region of the synaptic space) represent essentially the local contribution to the associated thermodynamic potential. For instance, (1) represents the local free energy contribution to the standard Helmholtz free energy

B Physically motivated pruning of a perceptron leads to a convolutional network

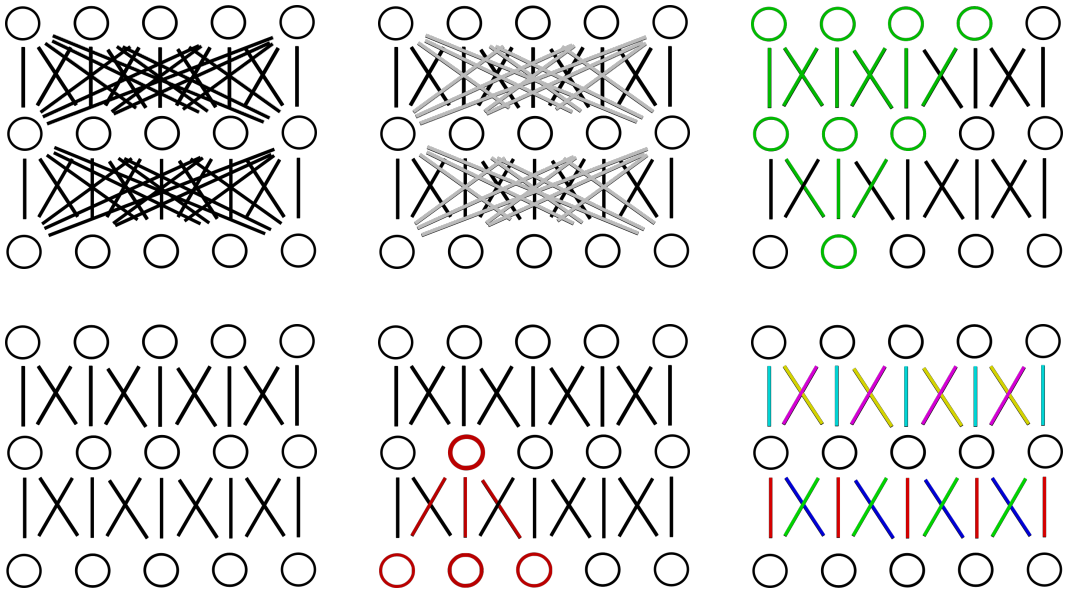

Figure 5: Convolutional architecture as a physically-inspired pruning of a fully-convolutional network.

In the main text, the inspiration coming from physics has been frequently appealed. Still resorting to physics, one can motivate the derivation of a convolutional architecture from a fully-connected one on the basis of generic principles. Specifically, locality and translational covariance.

This not only provides an organizing principle to "expect" that convolutional architectures may be convenient, it also provides a suggestive connection between the network and the geometry of spacetimes endowed with a light-cone structure.

Let us clarify by means of an explicit example. Consider a small fully-connected network formed by 15 neurons. For simplicity, take a cylindrical network, and consider three layers counting five neurons each, see Figure 5. Although having just 15 neurons, this multi-layer perceptron in its fully-connected configuration presents an already quite complicated link structure. It seems thereby natural to devise some simplified version, namely to find some criterion to reduce the number of links. This is sometimes referred to as a *pruning* procedure.

Suppose that the input layer is on the bottom of the pictures, so that the depth of the network coincides with height in the pictures of Figure 5. Also, associate the vertical direction (*i.e.* still the depth) to time, according to the logic that information flows from the input toward the output of the network. Now, assume that the "signals" have a finite horizontal propagation

speed. This means, for instance, that the output of a neuron A can reach only the neurons that are sufficiently close to the neuron lying on top of A, one layer deeper. Notice that this statement is introducing a specific notion of *locality* into the network. Actually, we are inducing a structure of light-cones, see Figure 5. Specifically, a neuron in the input layer can affect only those neurons which belong to a conical region above it, progressively widening when going deeper (*i.e.* as time progresses). Thus, the neurons belonging to this *future light cone* are the only ones which can be causally connected to the neuron sitting at the vertex of the cone.

Apart from the light-cone interpretation, note that locality has motivated a pruning technique of the fully-connected network which directly returned a "convolutional" structure. Actually, we are still half-way on our path to a convolutional architecture. To reach there we still need to comment translational symmetry.

Let us first observe that we can define also a *past light cone*. Namely, a neuron sitting at a point within the network can be influenced by all neurons which belong to a cone, progressively widening towards the input, whose vertex is the original neuron position. With different words, we have just re-expressed what is usually referred to as the neuron's *receptive field*.

The last, essential, ingredient to reach the convolutional network is related to symmetry, specifically, covariance of the network with respect to translational symmetry. Still referring to Figure 5, consider a pruned network where the weights connected by a horizontal (discrete) translation are constrained to be the same. The color coding in Figure 5 is meant to illustrate this idea. Now, combining the observation about the structure of the receptive fields and translational covariance, once can directly prove that the operation encoded in the neural network is actually a convolution, in the standard sense.

References

[1] C. Baldassi, A. Ingrosso, C. Lucibello, L. Saglietti, and R. Zecchina, "Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses," *Physical Review Letters*, vol. 115, Sep 2015.

[2] C. Baldassi, A. Ingrosso, C. Lucibello, L. Saglietti, and R. Zecchina, "Local entropy as a measure for sampling solutions in constraint satisfaction problems," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2016, p. 023301, Feb 2016.

[3] D. Musso, "Partial local entropy and anisotropy in deep weight spaces," *Phys. Rev. E*, vol. 103, no. 4, p. 042303, 2021.

[4] E. Witten, "A mini-introduction to information theory," *La Rivista del Nuovo Cimento*, vol. 43, p. 187–227, Mar 2020.

[5] T. Cover and J. Thomas, *Elements of Information Theory*. Wiley, 2012.

[6] O. Berezniuk, A. Figalli, R. Ghigliazza, and K. Musaelian, "A scale-dependent notion of effective dimension," 2020.

[7] C. Darken and J. Moody, "Towards faster stochastic gradient search," p. 1009–1016, 1991.

[8] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," p. III–1139–III–1147, 2013.

[9] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," 2017.

[10] A. M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B. D. Tracey, and D. D. Cox, "On the information bottleneck theory of deep learning," 2018.

[11] Z. Goldfeld, E. van den Berg, K. Greenewald, I. Melnyk, N. Nguyen, B. Kingsbury, and Y. Polyanskiy, "Estimating information flow in deep neural networks," 2019.

[12] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Journal of Machine Learning Research - Proceedings Track*, vol. 9, pp. 249–256, 01 2010.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," 2015.

[14] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," 2017.

[15] S. Amari, "Neural learning in structured parameter spaces - natural riemannian gradient," in *NIPS*, 1996.

[16] R. Pascanu and Y. Bengio, "Revisiting natural gradient for deep networks," 2014.

[17] T. Poggio, A. Banburski, and Q. Liao, "Theoretical issues in deep networks: Approximation, optimization and generalization," 2019.

[18] Q. Liao, B. Miranda, L. Rosasco, A. Banburski, R. Liang, J. Hidary, and T. Poggio, "Generalization puzzles in deep networks," 2020.

[19] C. Baldassi, E. M. Malatesta, and R. Zecchina, "Properties of the geometry of solutions and capacity of multilayer neural networks with rectified linear unit activations," *Physical Review Letters*, vol. 123, Oct 2019.

[20] L. N. Cooper and C. L. Scofield, "Mean-field theory of a neural network," *Proceedings of the National Academy of Sciences*, vol. 85, no. 6, pp. 1973–1977, 1988.

[21] M. Gabrié, "Mean-field inference methods for neural networks," *Journal of Physics A: Mathematical and Theoretical*, vol. 53, p. 223002, May 2020.

[22] M. Gabrie, *Towards an understanding of neural networks : mean-field incursions.* PhD thesis, 09 2019.

[23] S. S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein, "Deep information propagation," 2017.

[24] J. Erdmenger, K. T. Grosvenor, and R. Jefferson, "Towards quantifying information flows: relative entropy in deep neural networks and the renormalization group," 2021.

[25] C. Baldassi, C. Borgs, J. T. Chayes, A. Ingrosso, C. Lucibello, L. Saglietti, and R. Zecchina, "Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes," *Proceedings of the National Academy of Sciences*, vol. 113, p. E7655–E7662, Nov 2016.

[26] C. Baldassi, F. Pittorino, and R. Zecchina, "Shaping the learning landscape in neural networks around wide flat minima," *Proceedings of the National Academy of Sciences*, vol. 117, p. 161–170, Dec 2019.

[27] S. Zhang, A. Choromanska, and Y. LeCun, "Deep learning with elastic averaging sgd," 2015.

[28] S. Zhang, "Distributed stochastic optimization for deep learning (thesis)," 2016.

[29] S. E. Venegas-Andraca, W. Cruz-Santos, C. McGeoch, and M. Lanzagorta, "A cross-disciplinary introduction to quantum annealing-based algorithms," *Contemporary Physics*, vol. 59, p. 174–197, Apr 2018.

[30] Krauth, Werner and Mézard, Marc, "Storage capacity of memory networks with binary couplings," *J. Phys. France*, vol. 50, no. 20, pp. 3057–3066, 1989.

[31] A. Robins, "Catastrophic forgetting, rehearsal and pseudorehearsal," *Connection Science*, vol. 7, no. 2, pp. 123–146, 1995.