Statistical physics of neural systems

ScuDo

Scuola di Dottorato ~ Doctoral School

WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation

Doctoral Program in Physics (30^{th}cycle)

Statistical Physics of Neural Systems

By

Federica Gerace

******

Supervisor(s):

Prof. Riccardo Zecchina, Supervisor

Doctoral Examination Committee:

Prof. Guido Boffetta, Referee, University of Torino

Prof. Raffaella Burioni , Referee, University of Parma

Prof. Michele Caselle, University of Torino

Prof. Arianna Montorsi, Politecnico of Torino

Prof. Andrea Pagnani, Politecnico of Torino

Politecnico di Torino

2018

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Federica Gerace
2018

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

A mia Nonna per avermi indirizzato verso la Scienza

Acknowledgements

First of all, I would like to sincerely thank Riccardo Zecchina. There are no adequate words to express how grateful I am for the huge opportunity he gave me. He has been not only a supervisor but also a veritable guide, always there to lead us towards ambitious goals, to open our minds with his great intuitions and his way of performing research, and to encourage us in the worst of times, paying close attention to the different personalities of everyone and combining them in order to give rise to a cohesive and united team.

I am extremely grateful to Carlo Baldassi, who never backs down in providing explanations and in transferring his vast knowledge and his impressive computer skills, and to Carlo Lucibello, who believes to be my boss and who always provides original contributions, suggesting a lot of clever tricks.

I would like to thank the companion of many adventures Luca Saglietti, almost for everything but especially for having pushed me beyond my limits, by always trusting me, and Alessandro Ingrosso for his atypical but extremely useful advices and for his funny sense of humour. I would also like to thank Enzo Tartaglione, who is always ready to face new challenges with his great enthusiasm.

A special thank goes also to Carla Bosia, Anna Muntoni and Chiara E. Bena, who are living proof of how necessary is the contribution of women in the research world.

I would also like to thank Andrea Pagnani, for always trying to make me feel part of a group through his continuous mockeries, Andrea De Martino, Marco Del Giudice, Alfredo Braunstein, Luca Dall'Asta and all the other people in the lab with whom I had the pleasure of interacting and playing volleyball.

I am most grateful to my mother, my father, my brother and all my family for having always pushed me to do my best and to never give up, especially when life

severely strained my resolve. Finally, I would also like to thank all my friends, mainly Alessia, Andrea, Daniele, Federica, Marco, Marta, Paolo, Rosa and Stefano who never left me alone, not even for a second.

Abstract

The ability of processing and storing information is considered a characteristic trait of intelligent systems. In biological neural networks, learning is strongly believed to take place at the synaptic level, in terms of modulation of synaptic efficacy. It can be thus interpreted as the expression of a collective phenomena, emerging when neurons connect each other in constituting a complex network of interactions. In this work, we represent learning as an optimization problem, actually implementing a local search, in the synaptic space, of specific configurations, known as solutions and making a neural network able to accomplish a series of different tasks. For instance, we would like the network to adapt the strength of its synaptic connections, in order to be capable of classifying a series of objects, by assigning to each object its corresponding class-label. Supported by a series of experiments, it has been suggested that synapses may exploit a very few number of synaptic states for encoding information. It is known that this feature makes learning in neural networks a challenging task. Extending the large deviation analysis performed in the extreme case of binary synaptic couplings, in this work, we prove the existence of regions of the phase space, where solutions are organized in extremely dense clusters. This picture turns out to be invariant to the tuning of all the parameters of the model. Solutions within the clusters are more robust to noise, thus enhancing the learning performances. This has inspired the design of new learning algorithms, as well as it has clarified the effectiveness of the previously proposed ones. We further provide quantitative evidence that the gain achievable when considering a greater number of available synaptic states for encoding information, is consistent only up to a very few number of bits. This is in line with the above mentioned experimental results. Besides the challenging aspect of low precision synaptic connections, it is also known that the neuronal environment is extremely noisy. Whether stochasticity can enhance or worsen the learning performances is currently matter of debate. In this work, we consider a neural network model where the synaptic connections

are random variables, sampled according to a parametrized probability distribution. We prove that, this source of stochasticity naturally drives towards regions of the phase space at high densities of solutions. These regions are directly accessible by means of gradient descent strategies, over the parameters of the synaptic couplings distribution. We further set up a statistical physics analysis, through which we show that solutions in the dense regions are characterized by robustness and good generalization performances. Stochastic neural networks are also capable of building abstract representations of input stimuli and then generating new input samples, according to the inferred statistics of the input signal. In this regard, we propose a new learning rule, called Delayed Correlation Matching (DCM), that relying on the matching between time-delayed activity correlations, makes a neural network able to store patterns of neuronal activity. When considering hidden neuronal states, the DCM learning rule is also able to train Restricted Boltzmann Machines as generative models. In this work, we further require the DCM learning rule to fulfil some biological constraints, such as locality, sparseness of the neural coding and the Dale's principle. While retaining all these biological requirements, the DCM learning rule has shown to be effective for different network topologies, and in both on-line learning regimes and presence of correlated patterns. We further show that it is also able to prevent the creation of spurious attractor states.

Contents

Chapter 1

Introduction

Understanding how the brain is able to process and store information is one of the biggest challenges that the neuroscientific community is proposed to overcome. The research in this field takes on many facets and it is led at different scales.

The basic components of the brain are the neurons. They represent specialized cells able to quickly spread and share information over very large distances. To accomplish this task, neurons receive signals in the form of electric pulses, conveyed towards the centre of the cell, namely the soma, through extensions of the cell body, called dendrites. If the incoming stimulus is strong enough to make the membrane potential of the cell to exceed a certain threshold value, a neuron emits a sequence of electric pulses, that propagates across a long projection of the nerve cell, known as the axon.

The amplitude of each pulse, also called spike or action potential, is around 100 mV and it typically covers a time period of 1-2 ms [1]. Within a sequence of action potentials, or spike train, the time distance between one spike and another is established by the refractory period. During this interval of time, a neuron is completely insensible to whatever external stimulus, even the strongest one.

Neurons interact with the surrounding cells through the synapses. These are highly qualified structures that transmit the signal from the axon of a pre-synaptic neuron to the dendritic tree of the surrounding post-synaptic ones. In the synapses, the electrical signal is, most of the time, converted into a chemical one. Its transduction occurs through a series of complex bio-chemical stages, culminating in the release of special molecules: the neurotransmitters. These molecules are detected by specific

receptors that activate the opening of the ion-channels, located at the membrane of the post-synaptic cell. Their activation leads to a flux of ions from the extra-cellular medium to the inside of the cell, causing a change in the membrane potential of the cell itself [2].

Although still far from being exhaustive, the conquered knowledge about the biophysical and biochemical functioning of the nerve cells has inspired the design of several neural models, at various levels of accuracy. Whether to choose one model or another depends on the intended purpose.

For instance, many models aim to reproduce and to shed lights about the detailed functioning of single neurons. To this class belong the integrate-and-fire or the Hodgkin-Huxley models [3, 4]. These models typically end up in a series of coupled differential equations, through which they try to describe the fundamental electrical mechanisms undergoing within the nerve cells.

Conversely, there exist models of neurons embodying an oversimplified version of biological nerve cells: extremely complex and detailed models of single neurons are less prone to catch interesting phenomena, arising from the collective behaviour of an ensemble of nerve cells [5].

Indeed, it is hardly believed that processes like learning and memory emerge when neurons get connected with each other in creating a complex network of interactions. These collective phenomena are thus quite not affected by the details related to the single components of the network, in the same vein of what happens in other physical systems, like magnets, liquid crystals or superfluids [6].

The simplest neuronal model we can think of is the one of formal or artificial neurons, proposed by McCulloch and Pitts in 1943 and actually representing the first attempt in mathematically modelling neural networks [7]. Formal neurons constitute a caricature of biological nerve cells, here modelled as bistable linear thresholds elements, whose state is described through a binary variable $\boldsymbol{s}$, according to the fact that a neuron can fire or not.

Inputs from the external world, as well as, signals exchanged by the neurons within the network, cause the state of each neuron to change in time: a neuron decides to be in an active or in a quiescent state by weighting the incoming stimuli exerted by surrounding neurons, according to the strength of their synaptic connections, and then comparing their sum to a local threshold [6]:

$$s_i(t+1) = \phi\left(\sum_{j\neq i} W_{ij} s_j(t) - \theta_i\right), \tag{1.1}$$

where W_{ij} are known as synaptic couplings or weights, θ_i is the spiking threshold, while $\phi(x)$ represents some non linear activation function.

In this work, we will refer to this kind of neuronal model, of which we provide a qualitative drawing in Fig. 1.1.

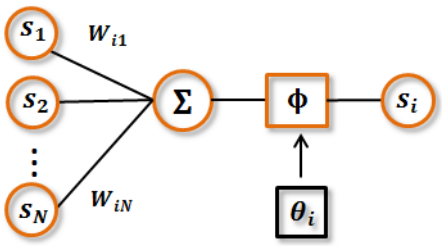

Fig. 1.1 Example of formal neurons. The incoming signal is first weighted (Σ) according to the strength of the synaptic couplings and then compared to a local threshold θ_i, by means of the non-linear activation function ϕ. This determines the neuronal state s_i [6].

When combined together, formal neurons give rise to different neural network topologies. In the discussions that follow, we will only consider the two harshest architectures, namely feed-forward and recurrent artificial neural networks. In particular, in the forthcoming sections of this first chapter, we will show how various methods borrowed from statistical physics, can come into play in the mathematical and physical investigation of feed-forward and recurrent neural networks, by exploiting the strong analogy between the physics of neural and disordered systems.

In the following chapters, we will instead present our work and our recent results related to both types of network architectures. In particular, in the second and third chapter, we will analyse the learning phenomenon in the simplest kind of feed-forward neural network: the Perceptron. In the fourth chapter, we will instead focus on the study of associative memory through recurrent neural networks, and network representations of the external stimuli. Finally, we will address conclusions, discussions and further perspectives to the fifth chapter. We remind to the Appendices for most of the technical details, related to both analytic calculations and numerical simulations.

1.1 Feed-Forward Neural Networks

A feed-forward neural network is an artificial neural network whose connections among neurons do not create cycles. It is typically organized in a sequence of $l = 1, ..., L$ neuronal layers, connected each other by a set of synaptic couplings $\mathbf{W} = \left\{ \mathbf{W}^{lk} \middle| 1 \leq l < L, 1 < k \leq L \right\}$, actually codifying the strength of the synaptic connections. Since no cycles are allowed, the input signal propagates forward from the input layer up to the output one: neurons belonging to the l-th layer only feed those in the $(l+1)$ layer, by processing the incoming stimulus exerted by the ones in the $(l-1)$ layer. According to Eq. 1.1, this simply means:

$$\begin{aligned} s_i^{l+1} &= \phi\left(\sum_{j\in l} W_{ij}^{l+1,l} s_j^l - \theta_i^{l+1}\right) \\ &= \phi\left(\sum_{j\in l} W_{ij}^{l+1,l} \phi\left(\sum_{j\in l-1} W_{ij}^{l,l-1} s_j^{l-1} - \theta_i^l\right) - \theta_i^{l+1}\right), \end{aligned} \tag{1.2}$$

with the non-linear activation function $\phi(x)$ being often chosen as the rectified linear function (ReLU): $\phi(x) = \max(x,\ 0)$ [8]. A qualitative drawing of a feed-forward neural network is shown in Fig. 1.2.

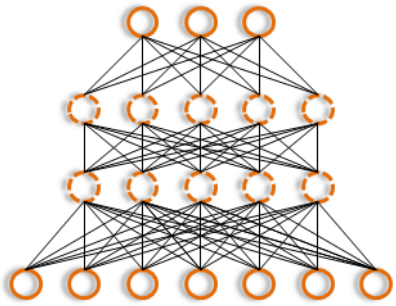

Fig. 1.2 Drawing of a feed-forward neural network. The thick and dashed circles represent the visible and the hidden neurons respectively. The input signal propagates from the bottom (input layer) to the top (output layer).

Neurons in the inner layers are often called *hidden* units, to distinguish them from those belonging to the input layer, usually known as *visibles*, given that their

state is completely clamped to the one imposed by the external stimulus. When multiple layers of hidden units are added, we refer to *deep* or *multilayer feed-forward neural networks*.

The architecture of a feed-forward neural network makes it particularly suitable in implementing *K-classification* tasks. In a K-classification task, the input signals are grouped into a number of K different classes, the network has then to correctly assign the corresponding class label to each one of the inputs. For instance, we can imagine to have at our disposal a set of images, each one depicting a different kind of animal. On the basis of these data, the network has to learn how to extract those aspects and features characterizing the animal, and that make it to belong to a given species rather than another one.

The features extraction is performed through the hidden layers: starting from the input data, from layer to layer, the network proceeds to build increasingly higher-level and more abstract representations of the input data, enhancing those traits of the data relevant for a good classification and neglecting useless details. For example, in the context of image classification, the first hidden layer usually specializes in detecting the edges of an image at particular positions and at given orientations. These are then usually combined by the subsequent hidden layers for the detection of more complex motifs (see Fig. 1.3) [8]. Currently, a turning point in image classification and recognition tasks is represented by a kind of deep feed-forward neural networks, known as *convolutional neural networks*, inspired by the organization of the neurons in the visual cortex [9].

The ensemble of methods and techniques enabling a machine to perform features extraction is known as *representational learning* [10]. In the case of a multi-layer feed forward neural network, when more than one level of representation of the input data is involved, we refer to *deep learning*.

A fundamental trait of deep learning is that the features related to a set of data are not given or fixed a priori. On the contrary, the network itself spontaneously and progressively learns how to extract them from the data. In the next section we will go more in details on that, by explaining how a neural network, trained according to specific learning protocols, can accomplish a given classification task.

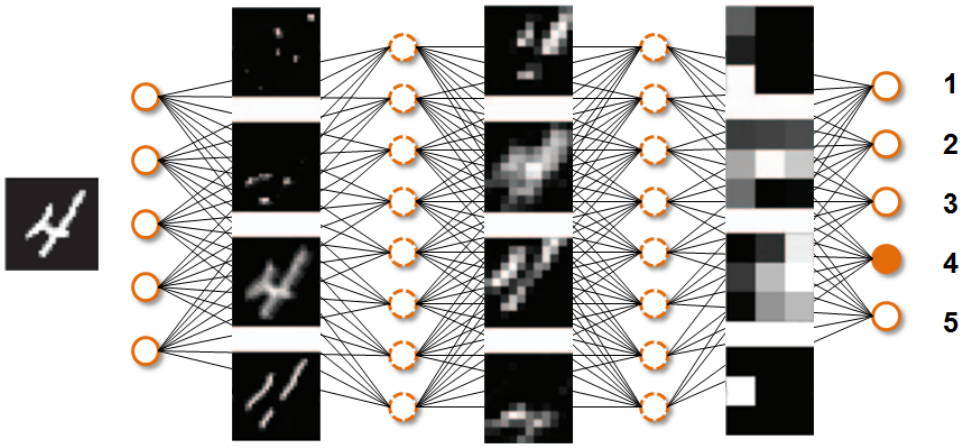

Fig. 1.3 Example of features extraction in feed-forward neural networks [10]. The input signal is represented by an hand-written digit in the MNIST dataset [11]. The network builds increasingly higher representations of the input data, allowing the network to recognize which number is represented in the image.

1.1.1 The Learning problem in classification tasks

Imagine that we have at our disposal a set of p input data or patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$. For example, they could be a series of grey-scale images featuring hand-written digits, as the ones collected in the MNIST data set [11], or a bunch of coloured images depicting different kinds of objects, like those gathered in the CIFAR data set [12]. Each one of the inputs is characterized by its own class-label $\{\mathbf{y}^{\mu}\}_{\mu=1}^{p}$, having K different classes.

This series of input data represents the *training set* and it is exploited by the neural network as a set of examples, through which the network learns how to detect useful features for a successful classification. A classification is successful if, given an input pattern $\boldsymbol{\xi}^{\mu}$, for instance a vector of pixels in the case of an image data, the response of the network is such that the input data is assigned to its corresponding class. The response is provided in the form of a vector of scores $\mathbf{s}^{\mu,L}$, one score for each class, readable as the configuration of the output layer and quantifying how likely is for the pattern to belong to a given class rather than another one. If the network has learnt to correctly classify the input data, then the index of the vector in correspondence with the highest score, coincides with the actual class of the input pattern [8].

This condition can be eventually reached only at the end of a training phase, during which the network adjusts its own parameters, namely the thresholds and the strength of the synaptic connections, in such a way to minimize a cost function E. This function quantifies a sort of distance between the response of the network and the desired one. Several cost functions can be defined, whether to choose one or another depends on the task we want to consider. For example, for a many-class classification problem we typically employ the categorical cross-entropy [10], namely $E\left(\mathbf{y}^{\mu},\mathbf{s}^{\mu,L}\right) = -\sum_{k=1}^{K} y_k^{\mu} \log\left(s_k^{\mu,L}\right)$.

In this way, the learning process simply translates into an optimization problem, whose *solutions* are represented by those configurations of the network parameters that minimize the training error, namely the number of wrongly classified patterns. In other words, the cost function defines an 'energy' landscape in the high-dimensional space of the network parameters, where the minima correspond to those configurations of the network satisfying a given classification task [8]. This is quite in analogy with a widespread belief in Neuroscience, according to which learning in the brain takes place in terms of modulation of synaptic efficacy [13–15].

In order to minimize the cost function, the strength of the synaptic couplings is progressively adapted by following the gradient of the cost function, a procedure known as *gradient descent* (GD) [16]:

$$W^{l+1,l} \leftarrow W^{l+1,l} + \eta \sum_{\mu=1}^{p} \frac{\partial E}{\partial W^{l+1,l}}, \tag{1.3}$$

with η representing the *learning rate* and actually quantifying the size of the steps of the descent. The gradient of the cost function is carried out by means of the chain rule, according to Eq. 1.2:

$$\frac{\partial E}{\partial W^{l+1,l}} = \frac{\partial E}{\partial s^{\mu,l+1}} \frac{\partial s^{\mu,l+1}}{\partial \phi} \frac{\partial \phi}{\partial W^{l+1,l}}. \tag{1.4}$$

Then, during the training phase, a pattern is presented to the input layer of a feed-forward neural network. The signal propagates forward up to the output layer, where the response of the network, in the form of a vector of scores, is collected. Then, the distance between the desired configuration of the output layer and the actual response of the network is computed through a cost function, that adds a penalty for every wrongly classified pattern. This generates a signal that propagates

backward through the layers of the network, modulating the strength of the synaptic connections in order to reduce the training error. The process is repeated for every pattern, until a minimum of the cost function is reached. This learning protocol is known as *back-propagation*, actually representing the core of such learning from examples or supervised learning [17].

One thing to notice here is that several variants of the standard GD algorithm are actually possible, one of them being the *Stochastic Gradient Descent* (SGD) [16, 18]. In this case, instead of summing up the contributions of all the p patterns, as shown in Eq. 1.3, the gradient is cumulated just on a small subset of the training set, called *batch*. The word 'stochastic' comes from the fact that the actual gradient is here replaced with a noisy estimation. The fluctuations thus introduced should help in reaching minima that can guarantee better classification performances [18].

The classification performances of the network can be checked at the end of the training on a bunch of input data that the network has never seen before, i.e. the *test set*, by looking at the *generalization error*, namely the equivalent of the training error on the test set.

The structure of the energy landscape is currently matter of a lively debate. We typically seek to minimize a non-convex cost function, characterized by many local minima. However, are all the minima accessible to current learning algorithms only local or can they find even global ones? Are they low-lying? Do flat minima provide better generalization performances than the sharper and isolated ones? Indeed it is believed that flat minima are more robust to noise and fluctuations and thus recently some efforts have been devoted in the design of GD-based algorithms that can effectively look for such regions [19].

As statistical physicists, we would like to investigate these problems and to shed lights on the actual functioning of deep-learning methods. In the next sections we will thus introduce the statistical physics approach to learning problems, especially focusing on the simplest kind of feed-forward neural network, the Perceptron, of which this work aims to provide more insights.

1.1.2 The statistical physics of learning from examples

The capability of inferring the underlying rule according to which a given input data is assigned to its corresponding class, is considered a characteristic trait of

intelligent systems. The fact that artificial neural networks can accomplish this task by learning from examples has significantly inspired the research on this field in statistical mechanics. Indeed, it is hardly believed that understanding the theory of learning starting from simple models, can provide more insights on the complex functioning of advanced intelligent systems.

Learning has always been a central topic in several disciplines, such as philosophy or psychology. However, thanks to the development of new tools for the analysis of complex systems, statistical physics has begun to provide its own contributions to the theory of learning from examples [20]. Despite the fact that it has only focused on quite simple models up to now, it has anyway proposed innovative points of view. Indeed, statistical physics has played a great role in the quantitative characterization of learning scenarios in simple learning systems, especially by pointing out the conditions under which good learning performances can be achieved. In this regard, great progress have been done by exploiting the strong analogy between learning and spin glass systems [21]. In this section we thus would like to briefly describe the approach of statistical physics to learning.

Statistical physicists do not look for bounds to the *worst* case, they are instead interested in characterizing the *typical* behaviour of a physical phenomenon. Similarly, in the theory of learning, we do not focus on the analysis of a specific learning scenario, emerging from a special choice of the training set. We are instead interested in the typical one. For this reason, we treat both the input patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$ and their associated class-labels $\{\mathbf{y}^{\mu}\}_{\mu=1}^{p}$ as random variables, drawn from some given probability distribution [22].

The synaptic couplings or, more generally, the parameters of the network represent the degrees of freedom of a learning system. Although their number can be consistently big, reaching hundreds of millions in deep networks [8], it is typically assumed to be of the order of the total number N of neurons within a network. Then, the limit $N \to \infty$ can be thus considered as a reasonable thermodynamic limit [6].

As the size of the network gets larger and larger, we expect the number of training examples p to scale as:

$$p = \alpha N \tag{1.5}$$

where α is a proportionality constant, known as *storage load* or *network capacity* [22]. Indeed, the more degrees of freedom we have the more constraints we need to fix them.

In the thermodynamic limit, *typical* does not simply mean 'most probable'. On the contrary, it also implies that the probability for every other not-typical event to occur, is negligible compared to a typical one. In this limit, we can thus identify some observables for which their typical value simply coincide with the averaged one, being their probability distribution sharply peaked around the typical value. Because of this property, these observables are called *self-averaging*: the average well represents the whole sample [23, 24].

In order to describe the typical learning scenario, the approach of statistical physics then consists in first identifying the self-averaging quantities, and second switching to the thermodynamic limit for computing their expectation values, well representing the typical behaviour. In the next section, we will concretely see how this strategy can be exploited, in order to extrapolate useful information about the theory of learning in Perceptrons.

1.1.3 The Perceptron

The simplest kind of feed-forward neural network is represented by the so called Perceptron. Introduced by Rosenblatt for the first time in 1961, high expectations have been suddenly placed in it for the development of artificial intelligence [25]. In fact, despite its simple structure, it has been proposed as a candidate model for catching some of the basic cognitive abilities of the brain.

A Perceptron is a single layer feed-forward neural network characterized by a set of N input neurons, whose state is described through binary variables $\{s_i\}_{i=1}^{N}$. The input layer is connected to a single binary output unit σ, through a set of synaptic couplings $\{W_i\}_{i=1}^{N}$. A sketch of such a network is shown in Fig. 1.4.

The Perceptron typically acts as a linear classifier: given a set of p input patterns $\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}$, representing specific configurations of the input layer, a Perceptron associates to each pattern a label, corresponding to one of the two possible states of the output unit, by performing a weighted sum of the incoming stimulus:

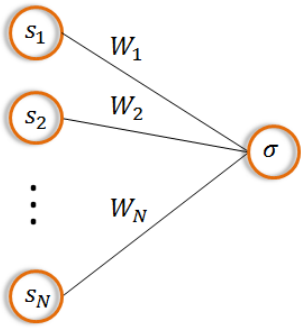

Fig. 1.4 Drawing of a Perceptron. The input layer is constituted by N input neurons, connected to a single output unit σ, through the synaptic couplings vector $\boldsymbol{W} = \{W_1, ..., W_N\}$.

$$\sigma^{\mu} = \phi\left(\sum_{i=1}^{N} W_i \xi_i^{\mu} - \theta\right) \qquad \forall \mu = 1, ..., p,$$

where θ is the spiking threshold, while $\phi(x)$ represents, as usual, some non-linear activation function. For the sake of simplicity, in the following we will choose as activation function the sign function: $\phi(x) = \text{sign}(x) = \{1 \text{ for } x > 0,\ 0 \text{ for } x = 0,\ -1 \text{ for } x < 0\}$. This does not represent a limitation for the purpose of the forthcoming discussions.

The set of input patterns is then automatically divided in two different classes: a pattern $\boldsymbol{\xi}^{\mu}$ belongs to the Ξ_+ (Ξ_-) class if it holds $\sigma^{\mu} = 1(-1)$. In particular, the condition:

$$\mathbf{W} \cdot \boldsymbol{\xi}^{\mu} - \theta = 0 \tag{1.6}$$

defines the hyper-plane in the high-dimensional space of the input patterns, at which the classification switches from one class to the other. The collection of points satisfying Eq. (1.6) represents the *decision boundary*.

As for generic feed-forward neural networks, given a set of p input patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$ and their corresponding class-labels $\{y^{\mu}\}_{\mu=1}^{p}$, the *learning problem* consists in inferring one or more possible configurations of the synaptic couplings, allowing a Perceptron to successfully assign to each one of the input patterns its

corresponding class-label. When this is the case, the response of the network σ^μ precisely matches the desired one y^μ.

This condition is eventually achieved at the end of a training phase, during which the Perceptron adjusts the strength of its synaptic connections, according to some given learning protocol, in order to reduce the training error. This quantity is thus exploited by the Perceptron as a feed-back signal, quantifying how many patterns have been wrongly classified up to a given point [26]:

$$\varepsilon_t\left(\boldsymbol{W};\{\boldsymbol{\xi}^\mu\}_{\mu=1}^p\right)=\frac{1}{p}\sum_{\mu=1}^{p}\Theta\left(-\sigma^\mu y^\mu\right), \tag{1.7}$$

where $\Theta(x)$ is the Heaviside step function: $\Theta(x)=\{1 \text{ for } x>0,\ 0 \text{ otherwise}\}$.

Unfortunately the training error does not represent a good indicator of the actual classification performances, since it is referred only to a subset of all the possible input patterns, namely the training set. As we have already pointed out, a better indicator is provided by the generalization error, viewed as the expectation over all possible patterns, namely not only the ones in the training set, of wrongly classifying a given input pattern [26]:

$$\varepsilon_g\left(\boldsymbol{W}\right)=\sum_{\{\boldsymbol{\xi}\}}P\left(\boldsymbol{\xi}\right)\Theta\left(-\sigma y\right). \tag{1.8}$$

The training error then represents just an estimation of the generalization error, in the same way as frequencies represent estimations of probabilities. We thus expect the training error to converge towards the generalization error, as soon as the number of training patterns p gets larger and larger.

According to how the set of class-labels $\{y^\mu\}_{\mu=1}^p$ is provided, we can distinguish between two main scenarios:

- the *teacher-student* or *generalization* scenario, where the class-labels are supplied through an underlying target rule g, that a *student* Perceptron has to infer: $y^\mu=g\left(\boldsymbol{\xi}^\mu\right)$. Typically the input-output association g is implemented by another Perceptron, usually called *teacher*;
- the *storage* or *classification* scenario, where the class-labels are chosen independently at random, according to a given probability distribution.

The initial excitement about the Perceptron started vanishing in 1969, when Minsky and Papert highlighted the limitations of such a neural network model [27]. Therefore, the research in this field did not proceed until the beginning of the 1980's. The turning point was represented by the work by Hinton on deep-networks and by Hopfield on recurrent neural networks, which we will discuss more in details in the second main section of this chapter.

However the Perceptron still represents the building block of the statistical mechanics of learning, as the hydrogen atom for the quantum world or the Ising model in statistical physics. It still constitutes an interesting and valuable model, on which both analytic calculation and numerical simulation can be carried out, as well as the starting point in the understanding of the more complex functioning of deep artificial neural networks.

In the following two sections, we will focus on two specific models of Perceptron: the continuous Perceptron and the binary Perceptron. We will present the approach of statistical physics to the analysis of both models, by showing the most relevant achievements, and, at the same time, we will briefly describe the learning protocols that are actually employed for training the two different topologies of Perceptrons.

Continuous Perceptron

The *continuous* Perceptron is a Perceptron model whose synaptic couplings are continuous variables, taking values on the real axis. This model has been the object of various studies, mainly because of its simplicity combined with the chance of an easy training by means of GD-based strategies, as we will see in the next section.

With no loose of generality, the vector of synaptic couplings is typically normalized to a sphere of radius $\sqrt{N}$:

$$\mathbf{W}^2 = \sum_{i=1}^{N} W_i^2 = N. \tag{1.9}$$

Making the vector lying on an N-dimensional sphere does not represent a limitation when the neuronal state variables are taken to be binary, and it determines a proper scaling in the thermodynamic limit. A continuous Perceptron satisfying this normalization constraint is known as *spherical* Perceptron.

In the forthcoming section we will analyse the learning problem in spherical Perceptrons, by means of methods and tools borrowed from statistical physics. The resulting approach goes under the name of *Gardner Analysis*. We will investigate both the teacher-student and the storage learning scenario, pointing out the theoretical scheme and the subsequent results.

The Gardner Analysis of spherical Perceptron models Statistical Physics turned out to be a powerful tool for the analysis of learning problems in Perceptron models. The solutions of a learning problem are represented by those configurations of the synaptic couplings, enabling a Perceptron to correctly classify a given set of input patterns. Learning can be thus seen as a dynamical and exploratory process across the N-dimensional synaptic space, that looks for those configurations of the synaptic couplings optimizing the training error.

As we have already pointed out, statistical physicists are interested in describing the typical learning scenario and not a specific realization, due to a special choice of the initial conditions and/or the training set. Therefore, both the input patterns and their associated class-labels are treated as random variables, sampled according to given probability distributions, respectively $P\left(\boldsymbol{\xi}^{\mu}\right)$ and $P\left(y^{\mu}\right)$.

Rephrasing the learning problem in the language of statistical physics, the synaptic couplings represent the microscopic variables of a learning system, leaving in the *phase space*. As the learning dynamics converges towards the equilibrium, the synaptic weights distribute in the phase space according to the Gibbs measure [22]:

$$P\left(\mathbf{W}\right) = \frac{1}{Z\left(\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}\right)} \exp\left(-\beta\varepsilon_{t}\left(\boldsymbol{W};\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}\right)\right), \tag{1.10}$$

where, as anticipated, the role of the energy is actually played by the training error. The normalization factor $Z\left(\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}\right)$ is known as the partition function and it is defined as a sum over all possible configurations of the synaptic couplings of the Gibbs measure:

$$Z\left(\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}\right) = \int d\mu\left(\mathbf{W}\right) \exp\left(-\beta\varepsilon_{t}\left(\boldsymbol{W};\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}\right)\right). \tag{1.11}$$

Here, $d\mu(\mathbf{W})$ represents the measure ensuring the spherical constraint to be satisfied:

$$d\mu(\mathbf{W}) = \frac{d\mathbf{W}\delta\left(\mathbf{W}^2 - N\right)}{\int d\mathbf{W}\delta\left(\mathbf{W}^2 - N\right)}. \tag{1.12}$$

In the low temperature limit, namely $\beta \to \infty$, the Boltzmann-Gibbs distribution is fully focused on the minima of the energy function, namely on the solutions of the learning problem. They occupy a region of the phase space known as the *version space*. In this limit, the partition function then simply translates into the volume of the version space, by restricting the integral over the synaptic couplings to the minima of the energy function:

$$\Omega\left(\boldsymbol{\xi}^{\mu}; y^{\mu}\right) = \int d\mu(\mathbf{W}) \prod_{\mu=1}^{p} \Theta\left(\sigma^{\mu}\left(\mathbf{W}, \boldsymbol{\xi}^{\mu}\right) y^{\mu}\right). \tag{1.13}$$

The volume of the version space is a quantity of interest in describing a learning scenario. Indeed, looking at how its size changes by varying the learning parameters, such as the storage load, it is possible to predict the specific conditions under which a learning problem switches from a satisfiability (SAT) phase, where one or more solutions can be detected, to an unsatisfiability (UN-SAT) one, when no solutions exist at all.

The volume of the version space represents a random variable itself, because defined through a product of randomly chosen training examples. In order to describe the typical learning scenario, we should thus average this quantity over the distribution of the training set, namely $P\left(\boldsymbol{\xi}^{\mu}\right)$ and $P\left(y^{\mu}\right)$. Unfortunately, the volume of the version space is not a self-averaging quantity: a product of random variables is known to be characterized by long tails probability distributions, for which the expected value do not coincide with the typical one [6].

A quantity that it is known to be extensive, and thus expected to be self-averaging in the thermodynamic limit, is the entropy, defined as the logarithm of the volume of the version space [28]:

$$S\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right) = \ln \Omega\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right). \tag{1.14}$$

Indeed, the entropy turns a product of random variables into a sum of random components. Because of the Central Limit Theorem (CLT), the distribution of a sum of an infinite number of random variables converges towards a Gaussian distribution, for which the mean coincides with the typical value:

$$S^{\text{typical}} = \left\langle \ln \Omega \left(\boldsymbol{\xi}^{\mu}, y^{\mu} \right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)} \tag{1.15}$$

The average over the training set, is *quenched* [22]: the time scales of the fluctuations introduced by randomly chosen training sets, are much bigger with respect to the ones characterizing the learning dynamics [29].

The computation of the quenched average is not straightforward. However, a series of pioneering papers has shown for the first time how to apply the *replica trick* to artificial neural networks in order to simplify this calculation [30–33]. In honour of their author, the resulting statistical physics approach goes under the name of *Gardner Analysis*.

The replica method has been largely exploited in the field of Spin Glass physics for dealing with quenched averages [24]. It is based on the trick of generating n different copies of the same system:

$$\left\langle \Omega^{n} \left(\boldsymbol{\xi}^{\mu}; y^{\mu} \right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)} = \left\langle \exp \left(n \ln \Omega \left(\boldsymbol{\xi}^{\mu}; y^{\mu} \right) \right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)} . \tag{1.16}$$

In the limit of the number of replicas going to zero, the dominant contributions can be considered the ones at most linear in n:

$$\left\langle \Omega^{n} \left(\boldsymbol{\xi}^{\mu}; y^{\mu} \right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)} \sim 1 + n \left\langle \ln \Omega \left(\boldsymbol{\xi}^{\mu}; y^{\mu} \right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu}; y^{\mu}\right)} . \tag{1.17}$$

Then, taking the logarithm of both the right and the hand side and considering once again only the leading contributions, it comes out the identity:

$$S^{\text{typical}} = \lim_{n \to 0} \frac{1}{n} \ln \left\langle \Omega^{n} \left(\boldsymbol{\xi}^{\mu}; y^{\mu} \right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)} . \tag{1.18}$$

The main advantage of the replica trick is that it allows to compute the average over the training set as first, thus applying the logarithm only afterwards. The exchange of the two operations considerably simplifies the computation of the

quenched average. The price we pay is that we have now to deal with n different copies of the same learning system.

In the next two sections, we will briefly describe the way of proceeding in computing the quenched entropy by means of the replica trick, in both the teacher-student and the storage learning scenario. We will then analyse the main results.

Teacher-Student scenario In the teacher-student or generalization scenario, a teacher learning system, typically a Perceptron, assigns to a set of p input patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$ their corresponding class-labels $\{y^{\mu}\}_{\mu=1}^{p}$. A student Perceptron has then to adjust its own synaptic couplings in such a way to infer the target rule:

$$y^{\mu} = g\left(\mathbf{T}, \boldsymbol{\xi}^{\mu}\right) \qquad \forall \mu = 1, ..., p, \tag{1.19}$$

according to which the teacher has divided the bunch of input patterns into two different classes, where $\boldsymbol{T}$ represents the teacher synaptic couplings vector.

In order to determine how close the student is in classifying the series of examples exactly as the teacher, it is typically introduced a parameter known as the *teacher-student* overlap:

$$R = \frac{\mathbf{W} \cdot \mathbf{T}}{N}. \tag{1.20}$$

Since both the teacher and the student coupling vectors lie on a N-dimensional sphere of radius $\sqrt{N}$, the teacher-student overlap precisely measures the cosine of the angle φ between the two vectors. Indeed, as it is shown in Fig. 1.5, where a projection in two dimensions of the N-dimensional sphere is provided, the teacher and the student couplings vectors identify two distinct decision boundaries, at which the classification of the input examples switches from one class to the other one. The region between the two planes then highlights the set of input patterns for which the response of the student, i.e. σ^{μ}, differs from that of the teacher, i.e. y^{μ}. Its size directly depends on the angle φ.

The generalization learning scenario has been intensively studied in the past through statistical physics [26, 34–37]. In this setting, the volume of the version space is given by the sum over all possible configurations of the student synaptic couplings, that classify the set of input patterns in agreement with the teacher:

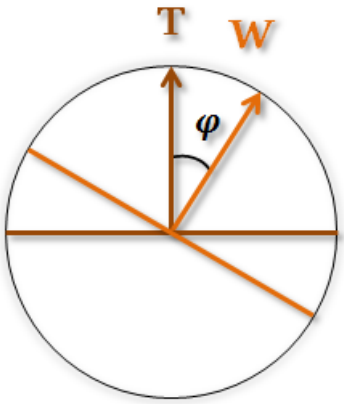

Fig. 1.5 Two dimensional sketch of the hyper-sphere on which the teacher and the student are confined. They select two different decision boundaries at which the classification switches from one class to another one. The region between the two boundaries is relative to those patterns for which the response of the student does not coincide with the one provided by the teacher [6].

$$\Omega\left(\boldsymbol{\xi}^{\mu},\mathbf{T}\right)=\int d\mu\left(\mathbf{W}\right)\prod_{\mu=1}^{p}\Theta\left(\left(\frac{1}{\sqrt{N}}\mathbf{T}\cdot\boldsymbol{\xi}^{\mu}\right)\left(\frac{1}{\sqrt{N}}\mathbf{W}\cdot\boldsymbol{\xi}^{\mu}\right)\right). \tag{1.21}$$

Taking advantage of the replica trick, the teacher-student quenched entropy can be written in terms of the logarithm of n different replicas of the volume of the version space:

$$S_{\text{t-s}}^{\text{typical}}=\lim_{n\to 0}\frac{1}{n}\ln\left\langle\Omega^{n}\left(\boldsymbol{\xi}^{\mu},\mathbf{T}\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu},\mathbf{T}\right)}$$

$$=\lim_{n\to 0}\frac{1}{n}\ln\left\langle\int\prod_{a=1}^{n}d\mu\left(\mathbf{W}^{a}\right)\prod_{a,\mu=1}^{n,p}\Theta\left(\frac{1}{N}\left(\mathbf{T}\cdot\boldsymbol{\xi}^{\mu}\right)\left(\mathbf{W}^{a}\cdot\boldsymbol{\xi}^{\mu}\right)\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu},\mathbf{T}\right)},$$

where both the training patterns and the teacher synaptic couplings are sampled according to the distributions:

$$P(\boldsymbol{\xi}) = \prod_{i=1}^{N} \left[\frac{1}{2}(\xi_i - 1) + \frac{1}{2}(\xi_i + 1) \right]$$
$$P(\mathbf{T}) = (2\pi e)^{-\frac{N}{2}} \delta\left(\mathbf{T}^2 - N\right), \tag{1.22}$$

thus assuming uncorrelated input examples and uniformly sampled teacher synaptic couplings on the N-dimensional sphere.

To proceed further, we need to constrain the teacher-student overlap and the overlap between different replicas of the student couplings vector, to the parameters R^a and q^{ab} respectively:

$$\delta\left(q^{ab}N - \mathbf{W}^a\mathbf{W}^b\right) = \frac{1}{2\pi}\int d\hat{q}^{ab}\, \exp\left(\hat{q}^{ab}\left(q^{ab}N - \mathbf{W}^a\mathbf{W}^b\right)\right)$$
$$\delta\left(R^aN - \mathbf{W}^a\mathbf{T}\right) = \frac{1}{2\pi}\int d\hat{R}^a \exp\left(\hat{R}^a\left(R^aN - \mathbf{W}^a\mathbf{T}\right)\right). \tag{1.23}$$

This allows to express the quenched entropy in terms of the integrals over the overlap parameters and their conjugates:

$$S_{\text{t-s}}^{\text{typical}} \sim \lim_{n\to 0}\frac{1}{n}\ln\int dq^{ab}d\hat{q}^{ab}dR^a d\hat{R}^a \exp\left(N\psi\left(q^{ab},\hat{q}^{ab},R^a,\hat{R}^a\right)\right). \tag{1.24}$$

The function $\psi(x)$ is typically called *action*. In the thermodynamic limit, the integrals can be carried out by means of the *saddle-point approximation*:

$$S_{\text{t-s}}^{\text{typical}} \sim N\lim_{n\to 0}\frac{1}{n}\left[\underset{q^{ab},\hat{q}^{ab},R^a,\hat{R}^a}{\text{extr}}\psi\left(q^{ab},\hat{q}^{ab},R^a,\hat{R}^a\right)\right]. \tag{1.25}$$

At the saddle point the conjugate overlap parameters, namely $\hat{q}^{ab}$ and $\hat{R}^a$ can be expressed as functions of the overlap parameters q^{ab} and R^a, so that the action depends explicitly only on the latter.

The action can be computed by assuming symmetry among replicas, namely $q^{ab} = q$ and $R^a = R$. The *replica symmetry* (RS) ansatz seems a reasonable assumption: on the basis of how replicas have been introduced, there is no reason to believe that each replica should behave in a different way with respect to all the others.

The RS ansatz turns out to be verified at the saddle point. However, as we will see, this is not always true when considering other kinds of models. When this is the case, the RS ansatz provides only an estimation of the action at the saddle point, whose stability has to be checked by means of perturbative techniques.

A further symmetry can be identified by noting that the teacher synaptic couplings are uniformly sampled in the weight space, according to the probability distribution $P(\mathbf{T})$. The same holds for the student synaptic couplings through the measure $d\mu(\mathbf{W})$. It is then natural to assume the student-student overlap to be indistinguishable from the teacher-student one: $R = q$.

Taking into account these assumptions, the action becomes a function of the solely teacher-student overlap:

$$S_{\text{t-s}}^{\text{typical}} \sim N \lim_{n\to 0} \frac{1}{n} \left[\operatorname*{extr}_R \psi(R) \right]$$

$$= N \operatorname*{extr}_R \left[\frac{1}{2}\ln(1-R) + \frac{R}{2} + 2\alpha \int \mathscr{D}t \, \mathrm{H}\left(-\sqrt{\frac{R}{1-R}}t \right) \ln \mathrm{H}\left(-\sqrt{\frac{R}{1-R}}t \right) \right], \tag{1.26}$$

with $\mathrm{H}(x) = \int_x^\infty \mathscr{D}t$ and $\mathscr{D}t = \left(dt/\sqrt{2\pi}\right) \exp\left(-t^2/2\right)$. The values of R optimizing the action have to be determined through the saddle-point equation:

$$R = \frac{\alpha}{\pi}\sqrt{1-R} \int \mathscr{D}t \frac{\exp\left(-Rt^2/2\right)}{\mathrm{H}\left(\sqrt{R}t\right)}, \tag{1.27}$$

with α being as usual the storage load. The saddle point equation thus relates the teacher-student overlap to the storage load of the network. When the storage load is equal to zero ($\alpha = 0$), the teacher student overlap vanishes ($R = 0$). Whereas, in the

limit of infinite storage loads ($\alpha \to \infty$), the teacher-student overlap converges to one ($R \to 1$).

Indeed, when no training examples have been yet shown to the network, every possible configuration of the student synaptic couplings can correctly classify a null number of input patterns. During the training, an increasing number of examples is presented to the network. This causes a shrink of the version space: those configurations that are unable to satisfy all the input patterns at once are progressively rejected, until only the teacher survives. In the teacher-student scenario we can thus identify the solely SAT phase, since the classification problem is always satisfied at worst by the teacher.

A plot of the teacher-student overlap as a function of the storage load, is shown in Fig. 1.6 (dashed line), together with the generalization error (solid line). In particular, the inset provides a plot of the teacher-student quenched entropy at the saddle-point. Its decreasing behaviour as a function of the storage load is a clear sign of the shrink of the version space.

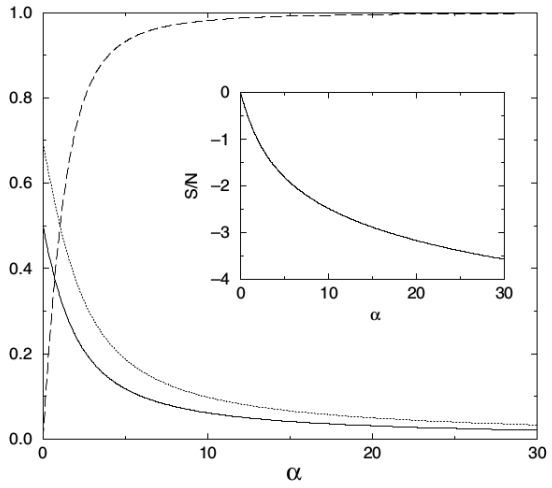

Fig. 1.6 Dashed line. Teacher-Student overlap R as a function of the storage load α. Solid line. Generalization error ε_g as a function of α. Inset. Quenched entropy density as a function of the network capacity α [6].

The Storage scenario In the Storage or Classification scenario, the class-labels $\{y^\mu\}_{\mu=1}^{p}$ are provided by sampling them at random from a given probability distribution $P(y^\mu)$. They are thus completely uncorrelated with the input patterns.

This learning scenario can be considered as an overly noisy teacher-student setting: the classification performed by the teacher is fully ruled by noise, making the role played by the teacher itself completely vain. In this context, the student has to be able to assign to each example its own class label, given that there is no correlation between the twos.

The aim of statistical physics is here to predict how many input patterns can be correctly classified by the student at the end of the training phase, thus determining the critical value of the storage load for which perfect classification is still possible.

The Gardner Analysis provides an answer to this question [30]. The starting point is once again the definition of the volume of the version space, written as a sum over all possible configurations of the synaptic couplings solving the learning problem:

$$\Omega\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right) = \int d\mu\left(\mathbf{W}\right) \prod_{\mu=1}^{p} \Theta\left(\frac{1}{\sqrt{N}} y^{\mu}\left(\boldsymbol{W}\cdot\boldsymbol{\xi}^{\mu}\right) - k\right), \tag{1.28}$$

where k represents the *stability parameter*, enhancing the requirement of successful classification. The replica trick comes then into play for the computation of the storage quenched entropy:

$$\begin{aligned} S_s^{\text{typical}} &= \lim_{n\to 0} \frac{1}{n} \ln \left\langle \Omega^n\left(\boldsymbol{\xi}^{\mu}; y^{\mu}\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)} \\ &= \lim_{n\to 0} \frac{1}{n} \ln \left\langle \int \prod_{a=1}^{n} d\mu\left(\mathbf{W}^a\right) \prod_{a,\mu=1}^{n,p} \Theta\left(\frac{1}{\sqrt{N}} y^{\mu}\left(\boldsymbol{W}^a\cdot\boldsymbol{\xi}^{\mu}\right) - k\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right)}, \end{aligned} \tag{1.29}$$

where both the training patterns and the class-labels are assumed to be identically and independently distributed:

$$P(\boldsymbol{\xi}) = \prod_{i=1}^{N} \left[\frac{1}{2}(\xi_i - 1) + \frac{1}{2}(\xi_i + 1) \right]$$
$$P(y^{\mu}) = \frac{1}{2}(y^{\mu} - 1) + \frac{1}{2}(y^{\mu} + 1). \tag{1.30}$$

As in the teacher-student learning scenario, the parameter q^{ab}, quantifying the overlap between two different replicas of the synaptic couplings vector, is introduced. At the saddle-point, assuming symmetry among replicas, namely $q^{ab} = q$, the action can be written as a function of the solely overlap parameter:

$$S_s^{\text{typ}} \simeq N \operatorname*{extr}_q \left[\frac{1}{2}\ln(1-q) + \frac{q}{2(1-q)} + \frac{\alpha}{2} \int \mathscr{D}t \, \ln \text{Erf}\left(\frac{k - \sqrt{q}t}{\sqrt{2(1-q)}} \right) \right], \tag{1.31}$$

where the value of q optimizing the quenched entropy has to be determined through the saddle-point equation:

$$\frac{q}{1-q} = \frac{\alpha}{\pi} \int \mathscr{D}t \exp\left(-\frac{\left(k - \sqrt{q}t\right)^2}{1-q} \right) \left[\frac{1}{2}\text{Erf}\left(\frac{k - \sqrt{q}t}{\sqrt{2(1-q)}} \right) \right]^{-2}. \tag{1.32}$$

Once again, the saddle point equation relates the overlap parameter to the storage load. When no training examples have been presented to the network yet, namely $\alpha = 0$, all configurations of the synaptic couplings are solutions of the learning problem, namely $q = 0$. Whereas, as the storage loads converges towards a critical value ($\alpha \to \alpha_c$), the overlap parameter goes to one ($q \to 1$), meaning that the version space has collapsed to a single point. Above the critical capacity, no more solutions to the learning problem can be found, as it is shown in Fig. 1.7. Therefore, in contrast to the teacher-student learning scenario, the Gardner Analysis here reveals the existence of a transition from a SAT to an UN-SAT phase.

Expanding the quenched entropy at the critical point, it is possible to determine the critical capacity as a function of the stability parameter:

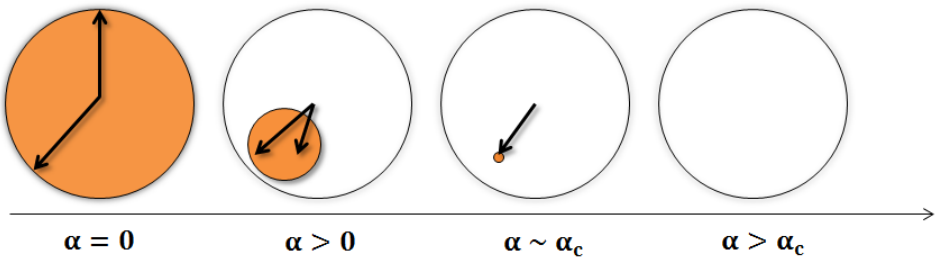

Fig. 1.7 Sketch of the shrink of the version space as a function of the storage load. The arrows describe two different couplings vectors $\boldsymbol{W}$. When the storage load α is equal to zero, whatever angle between the two vectors is allowed. As α starts increasing, the number of admissible angles starts getting smaller and smaller, up to the point where the two vectors simply collapse to a unique one [6].

$$\frac{1}{\alpha_c} = \int_{-\infty}^{k} \mathscr{D}t\,(k-t)^2\,. \tag{1.33}$$

As it is shown in Fig.1.8, the critical capacity is maximum when no stability is required, and it starts decreasing as soon as the stability parameter gets larger and larger. The depicted decreasing behaviour is something to be expected being the stability of a solution an extra requirement.

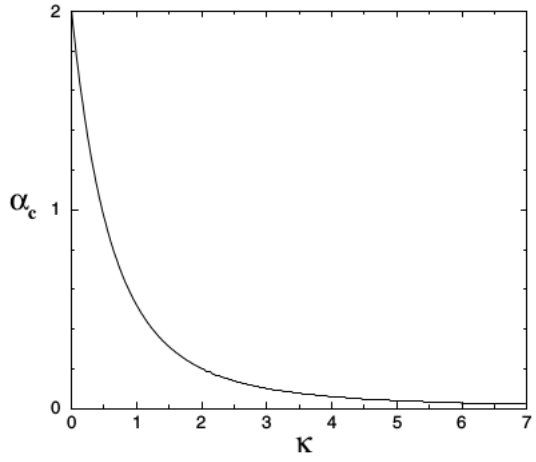

Fig. 1.8 Critical capacity α_c as a function of the stability parameter k [6].

In the next section we will see how these solutions can be concretely accessed through the design of suitable learning algorithms.

Learning algorithms for spherical Perceptrons Suppose to have a given set of p input examples $\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}$ together with their corresponding class-labels $\{y^{\mu}\}_{\mu=1}^{p}$. As said, a Perceptron learns how to assign to each example its own class label through a training process, during which the strength of the synaptic couplings is gradually modulated in such a way to optimize a given cost function.

The cost function represents a feed-back signal, providing information about how many examples have been wrongly classified up to a given stage of the training. As anticipated in the context of more complex architectures, its optimization can be carried out by means of GD-based strategies. These algorithms provide the recipe according to which the synaptic couplings have to be modified, in order to accomplish a given classification task. The recipe is typically called *learning rule*.

Depending on which cost function is chosen, it is possible to distinguish between different kinds of learning rules. However, for the purpose of the forthcoming discussions, we confine ourselves to a brief description of just two of the most popular ones: the *Perceptron* and the *Delta* learning rule.

The Perceptron learning rule has been introduced for the first time by Rosenblatt around the 1960s, becoming widely spread right after [25]. In this case, the synaptic couplings are modified only if the example to be classified has been assigned to the wrong class, otherwise they remain unchanged:

$$\mathbf{W} \longleftarrow \begin{cases} \mathbf{W} + \frac{\eta}{\sqrt{N}} y^{\mu} \boldsymbol{\xi}^{\mu} & \text{if } y^{\mu} \frac{\mathbf{W} \cdot \boldsymbol{\xi}^{\mu}}{\sqrt{N}} < 0 \\ \mathbf{W} & \text{otherwise.} \end{cases} \tag{1.34}$$

The update of the synaptic couplings has to be performed for every pattern and for a certain number of times, known as *epochs*, until all examples have been correctly classified.

The Perceptron learning rule can be derived by applying GD, as shown in Eq. 1.3, to the cost function:

$$E(\mathbf{W}) = \sum_{\mu=1}^{p} \left(-y^{\mu} \frac{\mathbf{W} \cdot \boldsymbol{\xi}^{\mu}}{\sqrt{N}} \right) \Theta \left(-y^{\mu} \frac{\mathbf{W} \cdot \boldsymbol{\xi}^{\mu}}{\sqrt{N}} \right), \tag{1.35}$$

which adds a penalty for every misclassified example. The advantage of this prescription is that it is quite easy to implement and it manages to find a solution to the classification problem in a reasonable number of epochs [6].

The Delta learning rule has been originally proposed by Widrow, the same time that the Perceptron learning rule made its first appearance [38]. As cost function, it resorts to the square error between the response of the network and the desired one [28]:

$$E(\mathbf{W}) = \frac{1}{2}\sum_{\mu=1}^{p}\left(y^{\mu} - \sigma^{\mu}(\mathbf{W})\right)^2 = \frac{1}{2}\sum_{\mu=1}^{p}\left(y^{\mu} - \phi\left(\frac{\mathbf{W}\cdot\boldsymbol{\xi}^{\mu}}{\sqrt{N}} - \theta\right)\right)^2, \tag{1.36}$$

being $\phi(x)$ a differentiable neuronal activation function. The optimization of the cost function by means of GD-strategies, leads to the Delta learning rule for the updates of the synaptic couplings:

$$\mathbf{W} \longleftarrow \mathbf{W} + \eta\left(y^{\mu} - \phi\left(\frac{\mathbf{W}\cdot\boldsymbol{\xi}^{\mu}}{\sqrt{N}} - \theta\right)\right)\phi'\left(\frac{\mathbf{W}\cdot\boldsymbol{\xi}^{\mu}}{\sqrt{N}} - \theta\right)\boldsymbol{\xi}^{\mu}, \tag{1.37}$$

with $\phi'(x)$ being the derivative of the activation function with respect to the synaptic weights.

In contrast to the Perceptron learning rule, this prescription provides for an adaptive increment of the synaptic couplings, depending on the deviation between the response of the network and the desired one. If compared to the Perceptron learning rule, where the increments are independent from the synaptic couplings, this guarantees a faster convergence of the rule in detecting a solution. Indeed, it has been shown that the squared error decreases exponentially with the number of epochs needed to reach a solution [39]. Another positive aspect is that it is possible to generalize the Delta rule to deep networks [6].

To summarize, we have shown how the Gardner Analysis can provide a quite comprehensive overview on the typical learning scenario, characterizing spherical Perceptron models. In particular, we have seen that, the teacher-student learning problem is always satisfied at least by the teacher, whereas, in the storage learning

scenario we can identify a critical value of the storage load, above which no more solutions to the classification problem can be detected. This determines a transition from a SAT to an UN-SAT phase, completely absent in the generalization scenario. In the SAT phase, the solutions to the learning problem can be concretely accessed through learning rules, based on GD strategies. We have described two of the most important ones.

In the next section we will instead focus on a different learning model: the binary Perceptron. We will see how the same statistical physics analysis can be set up by taking care of the due variations. Moreover, we will briefly describe the main algorithms exploited for training a binary Perceptron in accomplish a given classification task.

Binary Perceptron

The *binary* Perceptron is a Perceptron model, whose weights are allowed to take only two possible values, describing two different states of a synapse. It is equivalently called *Ising* Perceptron, in analogy to the well-known Ising model related to ferromagnetic systems.

Recent biological considerations and experimental evidence have suggested that synapses may act as binary switch devices, exploiting 1 to 5 bits per synapse for encoding information [40–42]. Then, despite the fact that synaptic couplings are typically assumed to be continuous, it would be relevant to start devising new learning models, where synapses are constrained to discrete values.

There are several reasons for which this kind of models are interesting to study: statistical physics analysis have shown that the critical capacity of binary networks is not consistently worse with respect to continuous models, discrete synaptic states are more robust to noise with respect to their continuous counterparts [6, 43, 44], they are also more suitable for machine learning applications. Binary networks are attractive even from a biological point of view: some experiments [40, 45], as well as theoretical arguments and numerical simulations [46–49], have pointed out that considering discrete synapses, rather than continuous ones, allows to reach long term information storage.

Training a binary network in solving a learning problem exhibits significant discrepancies with respect to the case involving continuous synaptic couplings. As

shown in the previous section, learning how to satisfy a given classification task in spherical Perceptrons can be easily tackled by means of GD-based strategies [6]. However, these algorithms can not be exploited in the context of binary networks, because of the discrete nature of the synaptic couplings. Learning is thus made much more difficult by the inherent structure of the binary model: a well known result has shown that training a binary Perceptron in solving a classification task represents an NP-complete problem in the worst case scenario [50, 51].

The origin of the computational hardness in training binary Perceptrons, has been clarified through statistical physics analysis, concerning the characterization of the typical learning scenario. These studies have shown that the energy landscape of a binary Perceptron model is characterized by many local minima in the thermodynamic limit [44, 52–54]. Usual local searching algorithms, based on Simulated Annealing and Monte Carlo strategies, are thus going to fail [55]. Moreover, a recent analysis, investigating the geometric properties of the version space, has revealed that solutions are typically isolated in binary Perceptrons models, making them even more hardly accessible by usual learning protocols [56].

Recently, more sophisticated algorithms, based on message-passing strategies, have been devised as learning protocols for the detection of those configurations of the synaptic couplings, enabling a binary Perceptron to solve a given classification task [57–60]. These algorithms have shown to be effective up to storage loads quite close to the theoretical bound, estimated by means of the Gardner Analysis, revised for the binary case [44].

The fundamental issue is how these newly proposed learning algorithms are actually able to detect a solution, despite the fact that typical solutions are isolated and embedded in an energy landscape characterized by exponentially many local minima. In the second chapter, we will try to explain why this is the case.

In the forthcoming sections, we will first show the relevant results emerging from a computation à la Gardner in binary Perceptron models. Then, we will briefly describe the above-mentioned learning protocols. The picture portrayed in the following sections will lay the basis for the discussions provided in the second and third chapter, where our direct contribution to the theory of discrete networks models is outlined.

The Gardner analysis of a binary Perceptron In order to shed lights on the typical learning scenario in binary Perceptron models, a powerful tool is as usual represented by the Gardner Analysis. The resultant theoretical scheme can be here set up qualitatively in the same way we have seen in the context of spherical Perceptron models.

In the next two sections, we will thus describe the main results concerning both the storage and the teacher-student learning scenario in binary Perceptron models. We will see to what extent the emerging picture deviates with respect to the continuous case, especially because of the binary nature of the synaptic couplings.

The storage scenario In the storage or classification scenario, the volume of the version space is defined as usual as a sum over all those configurations of the synaptic couplings, enabling a Perceptron to correctly classify a bunch of p input patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$, with a certain confidence k. The class-labels $\{y^{\mu}\}_{\mu=1}^{p}$ are here considered as identical and independently distributed random variables:

$$\Omega\left(\boldsymbol{\xi}^{\mu}, y^{\mu}\right) = \sum_{\{W_i=\pm 1\}} \prod_{\mu=1}^{p} \Theta\left(\frac{1}{\sqrt{N}} y^{\mu}\left(\boldsymbol{W}\cdot\boldsymbol{\xi}^{\mu}\right) - k\right). \tag{1.38}$$

Notice that, with respect to the spherical case of Eq. (1.28), the integral over the measure $d\mu\left(\mathbf{W}\right)$, and ensuring the spherical constraint to be satisfied, has been here replaced with a sum over all binary couplings.

The definition of the volume of the version space represents the starting point of the Gardner Analysis, leading to the estimate of the quenched entropy by means of the saddle-point approximation [6]:

$$S_S^{\text{typical}}\left(\alpha\right) \sim \underset{q,\hat{q}}{\text{extr}}\left[-\frac{\hat{q}}{2}\left(1-q\right) + \int \mathscr{D}z\, \text{ln2cosh}\left(\sqrt{\hat{q}}z\right) + \alpha \int \mathscr{D}t\, \text{lnH}\left(\frac{k-\sqrt{q}t}{\sqrt{1-q}}\right)\right], \tag{1.39}$$

where it has been assumed symmetry among replicas, namely $q^{ab} = q$ and $\hat{q}^{ab} = \hat{q}$, with q^{ab} being the parameter describing the overlap between two different replicas and $\hat{q}^{ab}$ its conjugate, as defined in Eq. (1.23).

The value of the overlap parameter and its conjugate optimizing the storage quenched entropy, have to be derived through the corresponding saddle-point equations [6]:

$$\begin{aligned} q &= \int \mathscr{D}z \tanh^2\left(\sqrt{\hat{q}}z\right) \\ \hat{q} &= \frac{\alpha}{2\pi(1-q)} \int \mathscr{D}t \exp\left(-\frac{\left(k-\sqrt{q}t\right)^2}{1-q}\right) \left[\mathrm{H}\left(\frac{k-\sqrt{q}t}{\sqrt{1-q}}\right)\right]^{-2}. \end{aligned} \tag{1.40}$$

Once again, the two equations establish the link between the overlap parameter and the storage load. They can be solved numerically, leading to the estimation of the critical capacity, at which the transition from the SAT to the UN-SAT phase takes place, namely $\alpha_c \simeq 1.27$ when no stability is required ($k = 0$) [44].

Indeed, the expansion of the quenched entropy in the limit of storage loads approaching the critical capacity, namely $\alpha \to \alpha_c$, predicts a critical value of the storage load ($\alpha_c = 4/\pi$) quite close to the one estimated numerically ($\alpha_c \simeq 1.27$) [44].

At first glance, this prediction may seem reasonable. Indeed, the Gardner Analysis of the storage scenario in spherical Perceptron models, estimates as maximum achievable capacity $\alpha_c = 2$ [61]. The very same analysis, performed instead in the context of binary Perceptron models, predicts the slightly lower value $\alpha_c \simeq 1.27$ [44]. This is something to be expected, being the binary Perceptron model a special case of the spherical one, where the synaptic couplings are constrained to binary values.

Unfortunately, this results is not reliable. Indeed, the amount of information needed to correctly classify a number $p = \alpha N$ of input patterns is encoded in the synaptic couplings, through their synaptic strength. Since the number of synaptic couplings in a Perceptron is exactly N, it means that there are at most N bits available for encoding the information carried by all the input patterns. Because of that, the critical capacity can never exceed one [6].

The reason why the Gardner Analysis leads to a wrong estimation of the critical capacity, has to be addressed to the improper assumption of symmetry among replicas. Indeed, analysing the stability of the RS saddle-point by means of local perturbations, it turns out that the RS assumption is verified at the saddle point

only up to $\alpha_s \simeq 1.015$ [44]. The wrongly predicted capacity is thus quite far away from the domain where the assumption of symmetry among replicas provides stable saddle-points.

The unreliability of the RS assumption takes its origin on the geometric properties of the version space [6]. Indeed, according to the theory of disordered systems, the assumption of symmetry among replicas typically describes quite well a system characterized by an ergodic dynamics [29]. Ergodicity implies a connected version space: given two possible solutions of the learning problem, there must always be at least one path joining the twos, without leaving the version space [6].

Unfortunately, contrary to the spherical case, the version space of a binary Perceptron model is not guaranteed to be connected [6]. The constraint of binary synaptic couplings makes the phase space to assume the structure of an N-dimensional hypercube, where each vertex represents one possible configuration of the synaptic couplings, including those representing a solution of the learning problem. Therefore, having a look at Fig. 1.9, two cases are possible: either two distinct binary solutions (filled dots) are connected through an edge of the hypercube, without leaving the version space of the spherical model (coloured region), as shown in (a), or they are not (b).

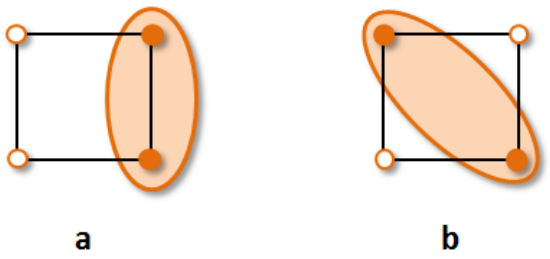

Fig. 1.9 Sketch in two dimensions of the hypercube denoting the geometry of the phase space for binary synaptic couplings. Each vertex of the hypercube represents a configuration of the synaptic couplings. In particular, the coloured vertices are the solutions of the binary learning problem. The coloured regions define the version space of the spherical Perceptron. a. Two solutions are connected by a path, without leaving the version space. b. The two solutions are disconnected, because there is no path joining the two without leaving the version space [6].

The solutions of the binary learning problem are thus spatially organized in different disconnected regions within the version space. Breaking the symmetry among replicas, is then necessary in order to make a distinction between configurations

of the synaptic couplings belonging to the same portion of the version space, and configurations belonging to different ones.

The simplest way to break the symmetry is to introduce two different parameters, namely q_1 and q_0, describing respectively the overlap between replicas within the same region of the version space and the one between replicas belonging to distinct ones [6]:

$$q^{ab} = \begin{cases} q_1 & 0 < |a-b| < m \\ q_0 & \text{otherwise.} \end{cases} \tag{1.41}$$

The same holds for the conjugate parameters:

$$\hat{q}^{ab} = \begin{cases} \hat{q}_1 & 0 < |a-b| < m \\ \hat{q}_0 & \text{otherwise.} \end{cases} \tag{1.42}$$

Here m represents a parameter counting the number of replicas that belong to the same region of the version space, with $m \in [1,...,n]$. In the limit of the total number n of replicas going to zero, m switches from an integer to a real number: $m \in [0,1]$, assuming the meaning of the probability for two distinct replicas to belong to two different portions of the version space [6].

The above symmetry assumption is usually known as *one-step replica symmetry breaking* (1-RSB) [24]. It leads to a slightly different expression for the quenched entropy, that takes into account the corrections due to the first level of symmetry breaking [6]:

$$\begin{aligned} S^{typ}_{1-RSB}(\alpha) = \underset{m,q_1,\hat{q}_1,q_0,\hat{q}_0}{\text{extr}} \Bigg[& \frac{m}{2}(q_0\hat{q}_0 - q_1\hat{q}_1) - \frac{\hat{q}_1}{2}(1-q_1) \\ & + \frac{1}{m}\int \mathscr{D}z_0 \ln \int \mathscr{D}z_1 \left[2\cosh\left(\sqrt{\hat{q}_0}z_0 + \sqrt{\hat{q}_1 - \hat{q}_0}z_1\right)\right]^m \\ & + \frac{\alpha}{m}\int \mathscr{D}t_0 \ln \int \mathscr{D}t_1 \mathrm{H}^m\left(-\frac{\sqrt{q_0}t_0 + \sqrt{\hat{q}_1 - \hat{q}_0}t_1}{\sqrt{1-q_1}}\right)\Bigg]. \end{aligned} \tag{1.43}$$

The value of the m and the overlap parameters optimizing the quenched entropy have to be determined, as usual, through their corresponding saddle-point equations. Close to the critical point, the saddle-point equations related to the overlap parameters depict a picture quite close to the one already met in spherical Perceptron models: the ensemble of the disconnected regions of solutions is expected to collapse to a single point ($q_1 \to 1$) when approaching the critical capacity, thus determining a shrink of the version space. Instead, in the same limit, the saddle-point equation associated to the m parameter, namely

$$0 = -\frac{m\hat{q}_0}{2}(1-q_0) + \frac{1}{m}\int \mathscr{D}z \ln 2\cosh\left(\sqrt{\hat{q}_0}z_0\right) + \frac{\alpha_c}{m}\int \mathscr{D}t \ln \mathrm{H}\left(-\frac{\sqrt{q_0}t}{\sqrt{1-q_0}}\right), \tag{1.44}$$

provides a criterion for the estimation of the critical value of the storage load [6]. Indeed, replacing q_0 and $\hat{q}_0$ with q and $\hat{q}/m$ respectively, the right hand side of Eq. (1.44) becomes exactly equal to the estimate of the quenched entropy under the RS assumption shown in Eq. (1.39). Therefore, it is not strictly necessary to face the more involved 1-RSB calculation, it is just sufficient to look for which value of the storage load, the RS estimate of the quenched entropy vanishes:

$$S_{RS}^{\text{typical}}(\alpha_c) = 0. \tag{1.45}$$

This criterion is reasonable when dealing with binary synaptic couplings. In this case, the quenched entropy exactly counts the number of solutions that can be found at a given value of the storage load [44]. Then, a vanishing entropy at the critical point is the clear sign of a shrink in the version space, where all configurations satisfying the classification problem are converging towards a unique one. The estimate of the critical capacity through the zero entropy criterion gives: $\alpha_c \simeq 0.83$, which is now consistent being smaller than one [44].

Teacher-student scenario In the teacher-student learning scenario, the volume of the version space is defined as the sum over all possible configurations of the student synaptic couplings, that classify a set of p input patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$ in agreement with the teacher:

$$\Omega\left(\boldsymbol{\xi}^{\mu},\mathbf{T}\right)=\sum_{\{W_i\pm 1\}}\prod_{\mu=1}^{p}\Theta\left(\left(\frac{1}{\sqrt{N}}\mathbf{T}\cdot\boldsymbol{\xi}^{\mu}\right)\left(\frac{1}{\sqrt{N}}\mathbf{W}\cdot\boldsymbol{\xi}^{\mu}\right)\right). \tag{1.46}$$

Starting from this definition, the Gardner Analysis provides, as usual, the theoretical scheme required for the estimation of the quenched entropy. Once again, this quantity can be derived under the assumption of symmetry among replicas at the saddle-point [6]:

$$\begin{aligned}S^{typ}\left(\alpha\right)=\underset{q,\hat{q},R,\hat{R}}{\text{extr}}\Bigg[&\frac{\hat{q}}{2}\left(q-1\right)-\hat{R}R+\int\mathscr{D}z\ln 2\cosh\left(\sqrt{\hat{q}}z+\hat{R}\right)\\&+2\alpha\int\mathscr{D}t\mathrm{H}\left(-\frac{Rt}{\sqrt{q-R^2}}\right)\ln\mathrm{H}\left(-\sqrt{\frac{q}{1-q}}t\right)\Bigg].\end{aligned} \tag{1.47}$$

As already pointed out in the context of spherical Perceptron models, the teacher-student scenario is characterized by the further symmetry $q=R$. This allows to take into account only the saddle-point equations relative to the teacher-student overlap and its conjugate respectively [6]:

$$\begin{aligned}R&=\int\mathscr{D}z\tanh\left(\sqrt{\hat{R}}z+\hat{R}\right)\\\hat{R}\sqrt{1-R}&=\frac{\alpha}{\pi}\int\mathscr{D}t\frac{\exp\left(-Rt^2/2\right)}{\mathrm{H}\left(\sqrt{R}t\right)}.\end{aligned} \tag{1.48}$$

As usual, the two equations establish the link between the teacher-student overlap and the storage load. Unfortunately, as seen in the storage scenario, even in this setting the RS assumption at the saddle point is not verified for every value of the storage load: the RS saddle-point optimizes the quenched entropy only up to $\alpha_d\simeq 1.245$, as shown in the inset of Fig. 1.10 [22]. Above α_d, the extreme value of the quenched entropy is no more the one obtained evaluating the quenched entropy at the RS saddle-point, but the one the quenched entropy assumes at the

boundary, namely $S^{\text{typical}}(R=1)=0$. This signals the existence of a first order phase transition from a poor ($R \neq 1$) to a perfect ($R=1$) generalization phase [22].

Because of the unreliability of the RS assumption, in principle, it should be necessary to introduce at least one level of symmetry breaking among replicas. However, the zero-entropy criterion introduced in the previous section, determines exactly α_d as critical capacity: Fig. 1.10 clearly shows that the RS estimate of the quenched entropy goes to zero exactly at the point where the assumption of symmetry among replicas is no more verified [22]. Therefore, despite the fact that the RS assumption does not provide stable fixed points beyond α_d, the instability occurs precisely at the point where the version space is dominated by the solely teacher. This allows to describe the entire learning scenario within the RS assumption [6].

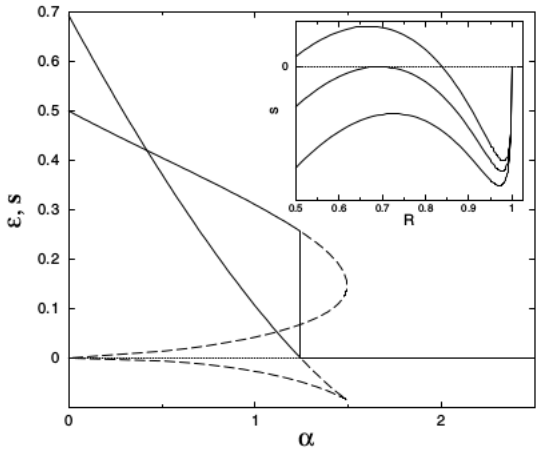

Fig. 1.10 Thin line. Quenched entropy as a function of the storage load α. Thick line. Generalization error as a function of the storage load α. The transition point $\alpha_c \simeq 1.245$ occurs precisely where the instability of the RS assumption (dashed lines) starts to appear. Inset. Quenched local entropy as a function of the typical teacher-student overlap parameter R, for increasing value of α. The quenched entropy is maximized by the RS saddle point only up to α_c. As α goes beyond the critical capacity, the maximum is provided by $R=1$ [6].

Learning algorithms for binary Perceptrons The Gardner Analysis sketched in the previous section, provides some hints on the geometric structure of the version space in binary Perceptron models: the need of breaking the symmetry among replicas suggests the picture of a fragmented version space, where the solutions to the classification problem are grouped in different disconnected regions.

As already pointed out in the description of the Gardner Analysis for spherical Perceptron models, the same analysis can be extended to the limit of finite temperatures, if considering the synaptic couplings to be distributed at equilibrium according to the Gibbs measure in Eq. (1.10) [22]. The resulting phase diagram in the α-temperature plane reveals the existence of a spin glass phase that, according to the theory of disordered systems, it is known to be characterized by a huge number of meta-stable states, whose energy barriers diverge with the system size [24, 44, 52–54]. .

As a consequence of that, the dynamics leading the network towards a minima of the cost function consistently slows down when crossing the spin glass phase. In binary Perceptrons, local searching algorithms are thus not effective as solvers of the learning problem: as soon as they enter the spin glass phase, they get easily trapped in the meta-stable states, thus being unable to reach a solution in a reasonable time [55].

Recently, a further analysis on the geometry of the version space has shown that the minima of the cost function, or, equivalently, the solutions to the binary learning problem, are typically isolated within the version space [56]. In binary Perceptron models, learning how to accomplish a give classification task is thus recognized to be an NP-complete problem [50, 51].

However, despite the fact that the picture emerging from the theoretical analysis is far from being encouraging, new effective learning algorithms, mostly based on *Belief Propagation* (BP), have been proposed over the last ten years. BP is a kind of message-passing algorithm: it computes the marginals related to the set of variables characterizing a physical system, through messages defined on factor graphs [62]. Notice that, in the case of learning systems, the variables are nothing but the synaptic couplings. Given the marginals, all the other physical quantities of interest, such as the free energy or the entropy, can then be estimated. A remind of how BP works is provided in Appendix A.

These newly proposed learning algorithms manage to detect a solution in a sub-exponential time, reaching critical capacities that are close to the theoretical bound of $\alpha_c \simeq 0.83$. In the following, we will briefly describe how they works.

The oldest one goes under the name of *Reinforced Belief Propagation* (R-BP) [57]. Along the lines of *message passing-guided decimation* strategies, this algorithm

turns BP into a solver for the binary classification problem, reaching an algorithmic critical capacity of $\alpha_c^{R-BP} \simeq 0.74$.

The decimation strategy is the one usually implemented for solving constraint satisfaction problems. It is divided in three different steps: first, it exploits a message-passing algorithm for estimating the marginals associated to the variables of a physical system; then, it takes advantage of the estimated marginals to set one of the variables to a specific value; finally, it tries to solve the constraint satisfaction problem, having fixed one of the variables to the given value. If this procedure does not produce any contradiction, the final configuration of the variables represents one of all the possible solutions of the constraint satisfaction problem [63].

R-BP can be meant as a kind of *smooth decimation* strategy in which *all* the variables are progressively and collectively fixed, until they reach a desired configuration. Fixing the value of a given variable, is equivalent to apply an infinite external field on it, that forces the variable to assume a specific value. Then, in R-BP, all the variables are subject to a time-dependent external field, whose function is to make them polarizing towards a solution of the problem [57].

The second learning algorithm, known as *Reinforced Max-Sum* (R-MS), is based on the same principle of R-BP but, instead of taking advantage of BP for the estimation of the marginals, it relies on the Max-Sum (MS) algorithm [58]. The latter can be conveniently viewed as the zero-temperature limit of the Belief Propagation algorithm. The reached critical capacity is here $\alpha_c^{R-MS} \simeq 0.75$.

With the purpose of designing a more biologically plausible learning algorithm, some simplifications of BP have been taken into account, giving rise to a third solver: the *Belief-Propagation Inspired* (BPI) learning algorithm [59]. In this case, the synaptic couplings are updated on-line, only according to information that are locally available to the synapses themselves. The resulting learning rule is very close to the Perceptron learning rule for spherical Perceptron models, although the two differ for two distinct reasons. First, BPI introduces a set of hidden variables, with the aim of modelling the internal states of each synapse. Second, it deals with situations in which the incoming signal is not consistently above or below threshold. BPI turns out to be very robust to noise. The biological requirements do not affect consistently the performances of the network, leading to a critical capacity of $\alpha_c^{BPI} \simeq 0.69$.

The last learning algorithm is known as *Clipped Perceptron Plus Reinforcement* (CP+R) [60]. It represents a further simplification and variation of the BPI algorithm:

the learning rule taking care of all those signals close to threshold, is here replaced by a generalized, stochastic and unsupervised reinforcement process of the synaptic couplings. This does not induce any additional improvement in terms of network capacity, being the critical storage load the same achievable through BPI, namely $\alpha_c^{CP+R} \simeq 0.69$.

To sum up, training a binary Perceptron in satisfying a given classification task, is harder with respect to the continuous case, where simple GD-based strategies can instead be adopted. We have seen how the approach of statistical physics to binary learning problems, can clarify the origins of such a computational hardness: the version space is disconnected and typical solutions are isolated and embedded in a landscape rich of meta-stable synaptic states. Nevertheless, we have also seen that there exist a bunch of recently proposed learning algorithms, that are actually effective in detecting solutions, approaching capacities quite close to the theoretical bound in reasonable time.

In the second and third chapter of the present work, we will try to solve this apparent contradiction, by showing how the previous described theoretical analysis were basically incomplete. We will see how a large deviation analysis can instead reveal the existence of extremely dense clusters of solutions, towards which we believe the recently proposed learning algorithms converge.

Up to now, we have seen how artificial neural networks can be exploited as discriminative models, where a bunch of different objects have to be assigned to their corresponding classes. However, artificial neural networks can actually be employed in a variety of different tasks. For instance, they can be exploited as memory devices, in order to store a set of patterns to be retrieved in due time, or they can be employed as generative models, in order to produce new samples according to the statistics of the training data. In the next section, we will thus provide an introduction to this different kinds of neural networks models, which represent the object of our recent work, proposed in the fourth chapter.

1.2 Recurrent Neural Networks

A *Recurrent Neural Network* is an artificial neural network, whose connections among neurons are allowed to generate cycles. In a cycle, the response of a neuron

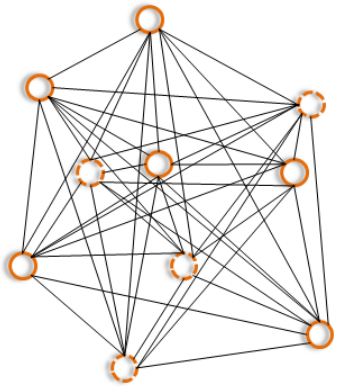

Fig. 1.11 Sketch of an highly recurrent neural network: each neuron is connected to all the others within the network. The thick circles are visible units while the dashed ones represent the hidden units.

plays the role of the incoming signal for all the other neurons to whom it is connected, thus giving rise to feedback mechanisms [64]. A sketch of an *highly* recurrent neural network is provided in Fig. 1.11. In this case, each neuron is connected to all the other neurons constituting the network. However, connections can be restricted to just a subset of all the neurons, as in the case of *diluted* recurrent neural networks.

The dynamics of a recurrent neural network is far to be as simple as the one already encountered in feed-forward neural networks. In the latter, the incoming signal simply propagates forwardly, from the input layer up to the output one. Due to the presence of feedback loops, the dynamics of a recurrent neural network can be instead rather complex: in principle, the network can chaotically explore all the possible configurations of the neuronal activity, without never reaching a stationary state [6].

Fortunately, this is not always the case: recurrent neural networks can exhibit an attractor dynamics for proper sets of synaptic couplings [6]. In this case, the dynamics of the network:

$$s_i(t+1) = \phi\left(\sum_{j\neq i} W_{ij} s_j(t) - \theta_i\right),$$

progressively evolves towards a configuration of the neuronal activity, that stays stable in time:

$$\xi_i = \phi\left(\sum_{j\neq i} W_{ij}\xi_j - \theta_i\right). \tag{1.49}$$

The pattern of activity $\boldsymbol{\xi}$ then represents an *attractor* state or, equivalently, a fixed point of the network dynamics: once the dynamics of the network reaches an attractor state, it will never run away from it.

Recurrent neural networks are typically employed in different contexts. For instance, they have been widely exploited as models of associative memory. In this case, the network store a bunch of p input patterns $\{\boldsymbol{\xi}\}_{\mu=1}^{p}$ as stable attractors of the network dynamics. Then, when triggered by a noisy or corrupted version of a given pattern, the network is able to retrieve the original one [64].

Statistical Physics turns out to be a powerful tool for shedding lights on recurrent neural networks as models of associative memory. It allows to answer questions like how many patterns is possible to store within a recurrent neural network, how the synaptic couplings have to be set in order to achieve the best storage performances, how much noisy a version of a given pattern can be for not preventing the network to retrieve the original one, what happens when the patterns are correlated and how much this correlation affects their retrieval. We will go through all these issues in the next section, entirely devoted to the Hopfield model, namely the first model of recurrent neural networks, that has been proposed for modelling associative memory.

Recurrent neural networks are further exploited as generative models. In this case, the network learns how to produce new samples of patterns, that preserve the same features of the ones on which it has been trained for accomplishing the task. We will deal with this topic in the section devoted to generative models. In particular, we will briefly introduce the Boltzmann Machines, a well-known kind of energy-based generative model.

1.2.1 The Hopfield Model

A turning point in the history of computational Neuroscience occurred at the beginning of the 1980s, when the physicist Jhon J. Hopfield proposed a model for associative memory [65]. The Hopfield model is considered as a milestone in the

field of neural networks. Indeed, it represents the first attempt of explaining how a memory can be retrieved, when triggered by an external stimulus.

In the Hopfield model, a recurrent neural network made of N binary neurons, namely $s_i \in \{-1,1\}$, is employed for storing a set of p patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^p$ as stable attractors of the dynamics, ruled by the Ising-like Hamiltonian:

$$H = -\frac{1}{2}\sum_{i,j=1}^{N} W_{ij}s_i s_j, \tag{1.50}$$

where the strength of the synaptic couplings is determined through the well-known Hebbian learning principle [66], according to which if two neurons are found to spike frequently together, their connection is reinforced, otherwise it is depressed:

$$W_{ij} = \frac{1}{N}\sum_{\mu=1}^{p} \xi_i^\mu \xi_j^\mu . \tag{1.51}$$

In 1985 Amit et al set up a statistical physics analysis of the Hopfield model, within the formalism of replicas [67]. This study has pointed out for which values of the storage load $\alpha = p/N$, the patterns to be stored can be identified with the minima of the Ising-like Hamiltonian of Eq. (1.50). The resulting phase diagram in the storage load - temperature plane is provided in Fig. 1.12a. The phase diagram clearly shows three different phases:

- the *ferromagnetic* phase (below T_c): the set of patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^p$ coincides with the global minima of the Ising-like Hamiltonian;
- the *spin-glass* phase (between T_c and T_M): the set of patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^p$ appears as meta-stable states of the Ising-like Hamiltonian;
- the *paramagnetic* phase (above T_M): the set of patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^p$ does not represent the minima of the Ising-like Hamiltonian.

Fig. 1.12b shows the number of wrongly retrieved patterns at $T = 0$, as a function of the storage load. As shown, perfect retrieval occurs only up to $\alpha = 0.055$. Above this value of the storage load, the number of errors starts progressively to increase, setting the critical capacity of the Hopfield model at $\alpha_c = 0.138$.

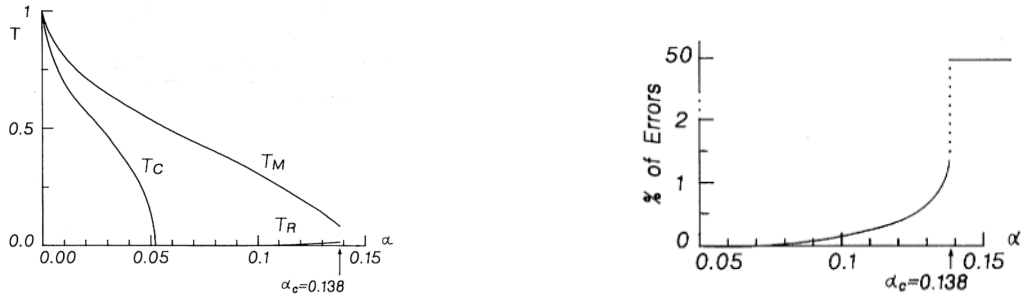

Fig. 1.12 Left. Phase diagram of the Hopfield model in the $\alpha - T$ plane. From left to right we can distinguish three different phases: ferromagnetic, spin glass and paramagnetic phase. The T_R line is relative to the region of the phase diagram below which replica symmetry braking is needed. Right. Generalization error as a function of the storage load α [67].

The Glauber Dynamics

The Hopfield model suggests the image of a network that moves from state to state in an energy landscape characterized by valleys and hills, till an attractor state, namely the bottom of a valley, is reached. The size of a valley is typically called *basin of attraction* [64]. Fig. 1.13 provides a sketch of the energy landscape of the Hopfield model in the ferromagnetic phase.

Modelling associative memory as an attractor dynamics, defined over an energy landscape, has been quite inspiring for physicists, being them used to the study of energy functions as well as the design of processes relaxing towards energy minima.

The attractor dynamics is typically simulated through a stochastic process that goes under the name of *Glauber Dynamics*. At each time, the state of every neuron is determined according to the Glauber transition probability [64]:

$$P\left(s_i^{t+1} \middle| \mathbf{s}^t\right) \propto \exp\left(\beta h_i^t s_i^{t+1}\right), \tag{1.52}$$

where h_i^t represents the neuronal local field, generally defined as the sum of two contributions, the external field and the field exerted by the surrounding neurons on neuron i, namely $h_i^t = h_i^{ext} + \sum_{j \neq i} W_{ij} s_j^t$. Note that, in the limit of $\beta \to \infty$, the Glauber dynamics translates into the deterministic dynamics of Eq. (1.1). The temperature then plays effectively the role of a noise source in the system.

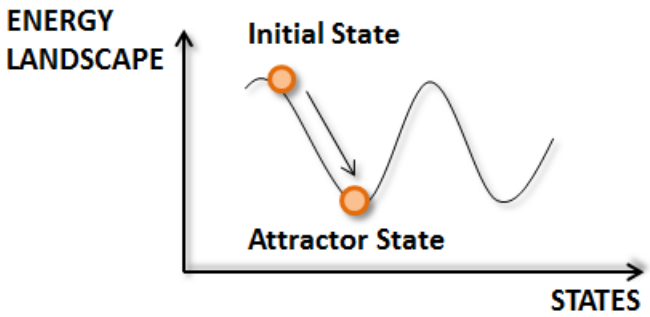

Fig. 1.13 Qualitative drawing of the energy landscape of the Hopfield model. If a pattern has been stored as a stable attractor of the network dynamics, then initializing the state of the network close to the pattern, it spontaneously converges towards the attractor state.

A pattern of activity is said to be properly stored within the network if, starting from a noisy version of it and performing a certain number of steps of Glauber dynamics, the configuration of the neuronal activity of the network converges towards the pattern itself.

In simulating the dynamics of the network, it is necessary to take care of the order according to which each neuron modifies its state in view of the Glauber transition probability. There are basically two different choices [64]:

- *synchronous dynamics*: all the state of the neurons are updated at once at time $t+1$, on the basis of the network activity at time t. In this case, it is like all the neurons follow a clock signal, after which they all update their state in parallel. This kind of dynamics is not completely reliable from a biological point of view: each neuron emits a spike whenever its membrane potential exceeds a certain threshold. However, this event is independent from the spiking of all the other neurons;

- *asynchronous dynamics*: the state of each neuron is updated one at a time, such that the update of every neuronal state relies on a configuration of the network activity where some of the neurons have already updated their state, while some others not yet. This dynamics is more biologically plausible, because it takes into account that signals among neurons propagate with different delays. Therefore, even if neurons may start firing synchronously, they will a-synchronize quite soon.

Limits of the Hopfield model

The Hopfield model and, in particular the Hebbian learning principle show several limitations. First of all, they are based on the two strong assumptions of both symmetric synaptic couplings ($W_{ij} = W_{ji}$) and full connectivity. These requirements are far to be realistic from a biological point of view. It is known that biological networks are sparse: each neuron is typically connected just to a subset ($\rho = 10^{-6}$) of all the others. Moreover, connections in biological networks are asymmetric: in principle, signals sent by neuron i to neuron j are never of the same strength of the ones sent by neuron j to neuron i [68].

The Hebbian learning principle allows to achieve the maximum storage load $\alpha_c = 0.138$, which is quite unsatisfactory. In particular, trying to add one more pattern, thus exceeding the critical capacity, the system enters in a glassy phase where, not only the newly proposed pattern can not be retrieved, but even the previously stored ones are completely forgotten by the network. This represents a well-known phenomenon that goes under the name of *catastrophic forgetting* [64].

Finally, the Hebbian learning rule as defined in Eq. (1.51), works just for unbiased independent and identically distributed random patterns. However, in more realistic and biological scenarios, the patterns of neuronal activity are instead correlated either spatially or semantically. One way to introduce correlations among patterns is to sample them from a biased probability distribution:

$$P\left(\xi_i^\mu\right) = f\,\delta\left(\xi_i^\mu - 1\right) + (1-f)\,\delta\left(\xi_i^\mu + 1\right), \tag{1.53}$$

with f being the bias [69]. In this case, the Hebbian leaning rule is not able to store biased patterns as stable attractors of the network dynamics, unless the rule is modified in light of a direct knowledge on the statistics of the patterns to store:

$$W_{ij} = \frac{1}{N}\sum_{\mu=1}^{p}\left(\xi_i^\mu - f\right)\left(\xi_j^\mu - f\right). \tag{1.54}$$

In the fourth chapter of the present work, we will go trough all these issues by proposing a new learning rule, able to overcome some of the limitations of Hebbian learning, while retaining a set of constraints that make a learning rule amenable from

a biological point of view. The newly proposed learning rule has been also tested in the context of generative models, which represents the main topic of the next section.

1.2.2 Generative Models

Feed-forward neural networks learn how to detect the relevant features of an object, in order to assign the object to its corresponding class. They thus typically act as *discriminative* models, devised for gathering sets of objects in different categories, on the basis of their characteristic traits. As seen in Sec. 1.1, to accomplish this task, the network is trained relying on a set of examples, made of pairs of input patterns and class-labels. This way of learning goes under the name of *supervised learning*.

Artificial neural networks can however be employed in satisfying a variety of different tasks, that go beyond object classification and that define new sets of machine learning techniques. As seen in the previous section, recurrent neural networks have been widely exploited as devices for the storage of patterns of neuronal activities. However, they are often also employed as *generative* models.

A generative model is a machine learning technique, designed for providing new data samples of the same kind of the ones constituting a training dataset. To accomplish this task, the model makes a guess on the unknown probability distribution according to which the data in the training set have been drawn, by building a complex parametrization of it. New data samples can then be generated, by directly sampling from the parametrized probability distribution. When this is the case, it is often said that the model has built an internal representation of the input data [70].

To fit the parameters of the guessed probability distribution, it is typically defined a suitable cost function, that has to be optimized with respect to the parameters of the distribution, in light of the training data. What suitable here means is not straightforward as in the case of learning in discriminative models: at the end of the training, the model is required to be able to catch complex correlations in the data structure and capable of good generalization performances.

Which cost function is the most fit for purpose is currently an open issue. However, there exists a wide variety of generative models, that takes advantage of the negative *log-likelihood* as suitable cost function for fitting the data. The negative log-likelihood is defined as the negative of the logarithm of the parametrized probability

distribution, according to which a set of p input data $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$, is supposed to be distributed:

$$\mathscr{L} = -\frac{1}{p}\sum_{\mu=1}^{p}\log\left(P\left(\boldsymbol{\xi}^{\mu};\{\lambda\}\right)\right), \tag{1.55}$$

with λ being one of the parameters of the guessed probability distribution and $P(x)$ being the likelihood, which, in statistical physics, coincides with the probability of having observed a set of realizations of a physical system, given the parameters of the model that is supposed to better describe the system [71].

Among all kinds of generative models, there exists a specific class known as *Energy-Based* generative model. In this case, the likelihood is assumed to be of the form of a Boltzmann-Gibbs probability distribution:

$$P\left(\boldsymbol{\xi}^{\mu};\{\lambda\}\right) = \frac{\exp\left(-\beta H\left(\boldsymbol{\xi}^{\mu};\{\lambda\}\right)\right)}{Z\left(\{\lambda\}\right)}, \tag{1.56}$$

with $H(x)$ being the Hamiltonian and $Z(x)$ the partition function, defined as a sum over all possible configurations of data samples, including the ones in the training dataset:

$$Z\left(\{\lambda\}\right) = \sum_{\{\boldsymbol{\xi}\}}\exp\left(-\beta H\left(\boldsymbol{\xi};\{\lambda\}\right)\right). \tag{1.57}$$

To determine the parameters of the distribution that best fit the data, the optimization of the corresponding negative log-likelihood is performed, as usual, by means of GD-based algorithms:

$$\lambda \leftarrow \lambda + \eta\frac{\partial\mathscr{L}}{\partial\lambda}. \tag{1.58}$$

In the case of energy-based generative models, the computation of the gradient of the negative log-likelihood leads to a matching condition between two distinct contributions:

$$\frac{\partial\mathscr{L}}{\partial\lambda} = \left\langle\frac{\partial H\left(\boldsymbol{\xi}^{\mu};\{\lambda\}\right)}{\partial\lambda}\right\rangle_{\text{data}} - \left\langle\frac{\partial H\left(\boldsymbol{\xi};\{\lambda\}\right)}{\partial\lambda}\right\rangle_{\text{model}}, \tag{1.59}$$

where $\langle \cdot \rangle_{\text{data}}$ represents the empirical average over the set of data, while $\langle \cdot \rangle_{\text{model}}$ denotes the expected value over the parametrized probability distribution. The two contributions are respectively called *positive phase* and *negative phase*.

The optimal configurations of the parameters of the guessed probability distribution, are then the ones that match the two phases, by lowering the energy of the samples that are close to the training data, and increasing the energy of the ones that instead deviate from them.

It is here important to notice that the estimation of the expected value is usually not straightforward. There are cases in which it can be estimated analytically (tractable likelihood), and others where instead it is necessary to resort to Monte Carlo samplings (intractable likelihood). Generative models significantly differ from discriminative models on this point: to optimize the cost function, the latter can take advantage of all the machinery of automatic differentiation through back-propagation.

In the next section we will briefly describe a specific kind of energy-based generative model, namely the Boltzmann Machine. In particular, we will focus on the special case of Restricted Boltzmann Machines.

Boltzmann Machines

A Boltzmann Machine is an example of energy-based generative model. It is constituted by a recurrent neural network where some of the neurons are visible, namely clamped on the input data, and some others are hidden, being their state not constrained a-priori to the one imposed by the input.

The presence of hidden units has the effect of increasing the representational power of the generative model. Indeed, the hidden units are left free to catch complex correlations in the data structure, thus allowing the network to construct an its own internal representation of the input data [72].

The reason why this is the case can be explained considering that, when hidden units are taken into account, the likelihood is defined by marginalizing over the hidden states:

$$P\left(\boldsymbol{\xi}^{\mu};\{W\}\right)=\sum_{\{\boldsymbol{h}\}}P\left(\boldsymbol{\xi}^{\mu},\boldsymbol{h};\{W\}\right)=\sum_{\{\boldsymbol{h}\}}\frac{\exp\left(\beta H\left(\boldsymbol{\xi}^{\mu},\boldsymbol{h};\{W\}\right)\right)}{Z\left(\{W\}\right)}. \tag{1.60}$$

The procedure of integrating out degrees of freedom gives rise to complex interactions among the visible units. This effect is nothing but the one observed, for example, in solid state physics where, tracing out the phonon degrees of freedom, has the effect of inducing complex interactions among the free electrons of the lattice [73].

In the context of Boltzmann Machines, the likelihood is parametrized by the synaptic couplings. The configuration of the synaptic couplings that best fit the input data is then determined by matching the positive and the negative phase:

$$W_{ij}\leftarrow W_{ij}+\eta\left\langle\frac{\partial F\left(\boldsymbol{\xi}^{\mu};\{W_{ij}\}\right)}{\partial W_{ij}}\right\rangle_{\text{data}}-\left\langle\frac{\partial F\left(\mathbf{v};\{W_{ij}\}\right)}{\partial W_{ij}}\right\rangle_{\text{model}}, \tag{1.61}$$

having defined $\mathbf{v}$ as the vector of visible units states and $F(x)$ as the free energy function, that takes into account the marginalization over the hidden states:

$$F\left(\mathbf{v};\{W\}\right)=-\log\left(\sum_{\{\boldsymbol{h}\}}\frac{\exp\left(\beta H\left(\mathbf{v},\boldsymbol{h};\{W\}\right)\right)}{Z\left(\{W\}\right)}\right). \tag{1.62}$$

However, as anticipated, while the positive phase simply represents an empirical average over the training data, the negative phase consists in an expected value over the parametrized distribution:

$$P\left(\mathbf{v};\{W\}\right)\propto\exp\left(-F\left(\mathbf{v};\{W\}\right)\right), \tag{1.63}$$

which can not be treated analytically. The negative phase needs then to be approximated by means of Monte Carlo Markov Chain (MCMC) importance sampling, which allows to estimate expected values as arithmetic averages, by sampling according to the desired probability distribution:

$$W_{ij} \leftarrow W_{ij} + \eta \left[\frac{1}{p} \sum_{\mu=1}^{p} \frac{\partial F\left(\boldsymbol{\xi}^{\mu}; \{W_{ij}\}\right)}{\partial W_{ij}} - \frac{1}{n} \sum_{s=1}^{n} \frac{\partial F\left(\mathbf{v}^{s}; \{W_{ij}\}\right)}{\partial W_{ij}} \right], \tag{1.64}$$

where n denotes the number of sampled configurations of the visible units.

Unfortunately, MCMC methods usually require very long mixing-times, making the training of a Boltzmann Machine intractable [74]. A simplified version of Boltzmann Machines is then typically taken into account. It goes under the name of Restricted Boltzmann Machine.

Restricted Boltzmann Machines A Restricted Boltzmann Machine (RBM) is a simplified version of a Boltzmann Machine, where only connections between visible and hidden units are allowed [75]. A sketch of an RBM is provided in Fig. 1.14.

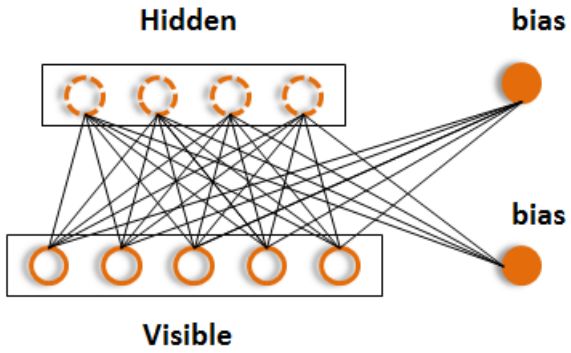

Fig. 1.14 Sketch of an RBM with only one layer of hidden units. The thick circles are the visible units while the dashed ones are the hidden units. The filled circles represent the biases.

In RBMs, the associated energy then acquires the Ising-like functional form:

$$H(\boldsymbol{v}, \boldsymbol{h}; \{W\}) = -b_v \sum_{i=1}^{V} v_i - b_h \sum_{i=1}^{H} h_i - \sum_{ij} W_{ij} v_i h_j, \tag{1.65}$$

where b_v and b_h represent the external fields, or bias, that can be applied on both visible and hidden units. The corresponding free energy is then obtained by marginalizing over the hidden states:

$$F(\boldsymbol{v};\{W\}) = -b_v \sum_{i=1}^{V} v_i - \sum_{\{\boldsymbol{h}\}} \log \left[\exp\left(-b_h \sum_{i=1}^{H} h_i - \sum_{ij} W_{ij} v_i h_j \right) \right]. \tag{1.66}$$

The configurations of the synaptic couplings that best fit the training data, are then determined according to the learning rule established by Eq. (1.64). The samples of configurations of the visible units $\{\boldsymbol{v}^s\}_{s=1}^n$ are typically extracted according to a kind of MCMC method, that goes under the name of *Gibbs Sampling*. Instead of sampling from the joint probability distribution of a set of random variables, this method approximate the joint distribution as the product of the marginals of the variables, conditioned to the value assumed by all the others, and then extract samples from this factorized distribution.

In the specific case of RBMs, because hidden-hidden and visible-visible connections are forbidden, all the visible units are independent, once the configuration of all the hidden units has been fixed. Therefore, in this case, the joint probability distribution simply coincides with the conditional factorized approximation:

$$P(\boldsymbol{v};\{W\}) = \prod_{i=1}^{V} P(v_i|\,\boldsymbol{h};\{W\}). \tag{1.67}$$

The same holds for the hidden units:

$$P(\boldsymbol{h};\{W\}) = \prod_{i=1}^{H} P(h_i|\,\boldsymbol{v};\{W\}). \tag{1.68}$$

It is thus possible to set up a Markov Chain, where, at each step of the chain t, the state of a given visible is sampled according to its corresponding probability distribution:

$$v_i^{t+1} \sim P\left(v_i^{t+1}\middle|\,\boldsymbol{h}^t;\beta,\{J,\theta\}\right). \tag{1.69}$$

Once again, the same holds for the hidden units:

$$h_i^{t+1} \sim P\left(h_i^{t+1}\middle|\,\boldsymbol{v}^t;\beta,\{J,\theta\}\right). \tag{1.70}$$

As already pointed out, MCMC methods require very long mixing-times. However, it has been shown that it is not necessary to estimate exactly the gradient of the negative log-likelihood, for making an RBM capable of generating an internal representation of the training data. What has been suggested is that even very few steps of Gibbs sampling can be sufficient. Moreover, some tricks can be exploited in order to make the convergence of the Gibbs sampling faster [76]. For example, initializing the Gibbs chain directly on the training data, biases the sampling, at the point that even one single step of the chain is sufficient for ensuring good performances:

$$W_{ij} \leftarrow W_{ij} + \eta \left[\left\langle \frac{\partial F\left(\boldsymbol{v}^1; \{W\}\right)}{\partial W_{ij}} \right\rangle_1 - \left\langle \frac{\partial F\left(\boldsymbol{\xi}^\mu; \{W\}\right)}{\partial W_{ij}} \right\rangle_0 \right]. \tag{1.71}$$

Here, the subscript 1 refers to the unique sample extracted from the first step of the Gibbs chain while the subscript 0 refers to the initial state of the Gibbs chain, clamped on the training data. The resulting update rule is known as *Contrastive Divergence* (CD) and represents the current state-of-the-art in RBMs training [77].

Chapter 2

The Discrete Perceptron

The first section of the first chapter provides an overview on learning in feed-forward neural networks. Their architecture makes this topology of artificial neural network particularly suitable as discriminative model. In this case, the network is required to classify a set of objects, by assigning to each object its corresponding class-label. This condition is eventually reached at the end of a training phase, where the synaptic couplings are progressively modulated, in order to optimize a given cost function carrying information on the number of wrongly classified objects.

Learning how to accomplish a classification task, then simply translates into an optimization problem, whose solutions are represented by those configurations of the synaptic couplings enabling the network to minimize the number of wrongly classified patterns.

We have shown how statistical physics can come into play, in order to shed lights on the typical learning scenario. In particular, we have pointed out how methods borrowed from the physics of disordered systems, can extrapolate useful information on learning in the simplest kind of feed-forward neural network, namely the Percetron. We have especially focused on the binary Perceptron model, namely when only two synaptic states are available for encoding information. We have seen that, the analysis of the typical learning scenario in binary Perceptron models suggests the picture of a disconnected space of solutions, where exponentially many typical solutions are isolated and embedded in a landscape rich of meta-stable synaptic states.

Despite the fact that the theoretical analysis depicts a scenario where solutions are typically hard to find, we have seen that a set of recently proposed learning algorithms have shown to be effective in solving the binary Perceptron learning problem. These solvers work reasonably well, even in their simplified formulation and even when trying to be more conceivable from a biological point of view.

Why this is the case is the central question which we will investigate in this second chapter. We will show how the previous theoretical analysis were basically incomplete. In particular, we will propose a large deviation analysis that can reveal the existence of extensive and extremely dense clusters of solutions, towards which the above-mentioned learning algorithms seem to converge.

The existence of sub-dominant and extremely dense clusters of solutions has been already shown to be a characteristic feature of binary Perceptron models [78]. Here, we want to extend both the typical analysis in Ref. [56] and the large deviation analysis in Ref. [78] to the more general case of discrete Perceptron models, namely when the number of available synaptic states lies on a finite and discrete range. We sometimes refer to this model as the Potts Perceptron, in analogy to the well-known Potts model in statistical mechanics. In particular, we will show how the above-mentioned scenario still holds in this more general setting, even when taking into account some relevant biological constraints, like the Dale's principle or the sparseness of the neural coding.

This chapter is divided in four different sections. In the first section, we introduce the discrete Perceptron learning model. In the second section, we present the main results emerging from the analysis of the typical learning scenario in discrete Perceptron models. In the third section, we set up the large deviation analysis aiming to shed more lights on the complex structure of the solution space. Finally, in the fourth section, we propose a new learning algorithm for discrete Perceptrons, that goes under the name of Entropy Driven Monte Carlo (EDMC). The technical details related to the analytic calculation sketched in the present chapter can be found in Ref. [79].

2.1 The model

The *discrete* or *Potts* Perceptron is a Perceptron model, whose synaptic couplings are constrained to take values in a finite and discrete range. For the sake of simplicity, we assume the synaptic states to lie in the range $W_i \in \{0,1,...,L\}$. This assumption does not affect the final result, being our analysis general and thus independent from the specific choice of the synaptic states.

As anticipated, we want to investigate the learning scenario in discrete Perceptron models, while retaining some biological constraints. First of all, we need to consider that biological networks are characterized by sparse neural coding. In the following, we will thus consider the neuronal states to be described by $s_i \in \{0,1\}$ binary variables, thus avoiding the unrealistic feature of symmetry between the active and the quiescent state of the $s_i \in \{-1,1\}$ binary model. Second, the synaptic couplings are required to satisfy the Dale's principle, according to which synaptic connections are either excitatory (positive synaptic couplings) or inhibitory (negative synaptic couplings) [80, 81]. The chosen range of available synaptic states then fits this requirement.

In a classification task, the Perceptron is required to correctly assign to a set of p input patterns $\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}$ their corresponding class-labels $\{y^{\mu}\}_{\mu=1}^{p}$. These defines the training set. As anticipated, in statistical physics, we are not interested in a specific realization of the learning scenario, dictated by a special choice of the training set, we are instead interested on the typical case. We thus treat both the input patterns and their associated class labels as independent, identically and biased distributed random variables:

$$\begin{aligned} P\left(\boldsymbol{\xi}^{\mu}\right) &= \prod_{i=1}^{N}\left[f\,\delta\left(\xi_i^{\mu}-1\right)+(1-f)\,\delta\left(\xi_i^{\mu}\right)\right] \\ P\left(y^{\mu}\right) &= f\,\delta\left(y^{\mu}-1\right)+(1-f)\,\delta\left(y^{\mu}\right) \end{aligned} \quad , \tag{2.1}$$

where the bias f specifies the level of sparseness of the neural coding, therefore it is typically known as *sparsity* or *coding level*. In principle, we could consider two distinct coding levels for the patterns and the class-labels, however this represents

an unnecessary complication and it does not constitute a remarkable detail for the following discussion.

The Perceptron decides whether to assign an input pattern to one class or another, by performing a weighted sum of the incoming stimulus and comparing it to a local threshold:

$$\sigma^{\mu} = \Theta\left(\sum_{i=1}^{N} W_i \xi_i^{\mu} - \theta N\right) \qquad \forall \mu \in \{1, ..., p\}\,. \tag{2.2}$$

The learning problem then consists in determining those configurations of the synaptic couplings, enabling the Perceptron to minimize the number of wrongly classified patterns, namely those patterns for which the response of the network σ^{μ} does not coincide with the actual class-label y^{μ}. The learning problem is thus translated into an optimization problem, whose solutions are represented by those configurations of the synaptic couplings minimizing the training error:

$$E(\mathbf{W}) = \sum_{\mu=1}^{\alpha N} \Theta\big(-(2\sigma^{\mu} - 1)(2y^{\mu} - 1)\big). \tag{2.3}$$

This optimization problem can be equivalently viewed as a constraint satisfaction problem, where each pattern imposes a constraint in the search of the optimal configuration of the synaptic couplings. The optimum will then be represented by the configurations of the synaptic couplings that satisfies all the constraints at once:

$$\mathbb{X}_{\xi,y}(\mathbf{W}) = \prod_{\mu=1}^{\alpha N} \big(1 - \Theta\big(-(2\sigma^{\mu} - 1)(2y^{\mu} - 1)\big)\big) = 1. \tag{2.4}$$

According to how the class-labels are provided, we know we can basically distinguish between two different learning scenarios: the teacher-student (or generalization) and the storage (or classification) scenarios. Despite the fact that the teacher-student scenario is more interesting than the storage one, in this work we will only consider the latter. Indeed, it has already been shown that, the picture emerging from the large deviation analysis of the storage scenario in binary Perceptrons, remains qualitatively unchanged in the teacher-student setting [58]. We thus reasonably believe that this is also the case when considering the generalization to multi-valued synaptic couplings.

2.2 The Typical Analysis

In order to investigate the typical learning scenario in discrete Perceptron models, the starting point is as usual represented by the Gardner Analysis. As seen, the ultimate goal of the Gardner Analysis is the estimation of the quenched entropy as the average over the training set of the volume of the version space:

$$S^{\text{typ}} = \frac{1}{N} \left\langle \log \sum_{\{\boldsymbol{W}\}} \mathbb{X}_{\xi,y}(\boldsymbol{W}) \right\rangle_{P(\boldsymbol{\xi}^{\mu}, y^{\mu})} . \tag{2.5}$$

As seen, in the case of discrete models, the quenched entropy directly counts the number of configurations of the synaptic couplings solving the learning problem. Computing this quantity through the formalism of replicas then leads to the estimation of the critical value of the storage load, that determines the transition from the SAT to the UN-SAT phase.

In the next section, we will show the relevant results emerging from the Gardner Analysis. In particular, we will see how the critical capacity is modified by tuning the parameters of the model, namely the coding level and the number of available synaptic states.

2.2.1 Critical storage load

The Gardner Analysis of the storage learning scenario in binary Perceptron models, has revealed the existence of a critical value of the storage load, namely $\alpha_c = 0.833$, at which the learning system switches from a SAT phase, where more than one solutions to the learning problem can be detected, to an UN-SAT phase, where instead no solutions can be found at all.

This sharp transition is actually a characteristic trait of Perceptron models with discrete synaptic couplings. A similar Perceptron model, with two available synaptic states ($L = 1$) and unbiased training set ($f = 0.5$), is characterized by the same discontinuous learning behaviour, with a critical value of the storage load $\alpha_c = 0.59$ [43]. When $L \to \infty$, the discrete Perceptron model reduces to a continuous Perceptron model, where the synaptic couplings are constrained to satisfy the Dale's law. In this case, the critical capacity has been shown to be $\alpha_c = 1$ [82], which is consistent

with the maximum achievable capacity $\alpha_c = 2$, predicted by the Gardner Analysis for spherical Perceptron models.

In the following, we show how the critical value of the storage load behaves as a function of a generic coding level f and of a generic number of available synaptic states $L+1$. Having a look at Fig. 2.1A, the sparser the neural coding is, the more training patterns the network is able to successfully classify. Indeed, sparse neural codings are capable of encoding greater amount of information [83]. Similarly, the critical capacity increases with the number of available synaptic states, having the network at its disposal a greater number of bits for encoding information. Moreover, the behaviour of the critical capacity as a function of the number of synaptic states, looks quite the same for lower coding levels. These results are in accordance with the ones obtained in a slightly different scenario [43].

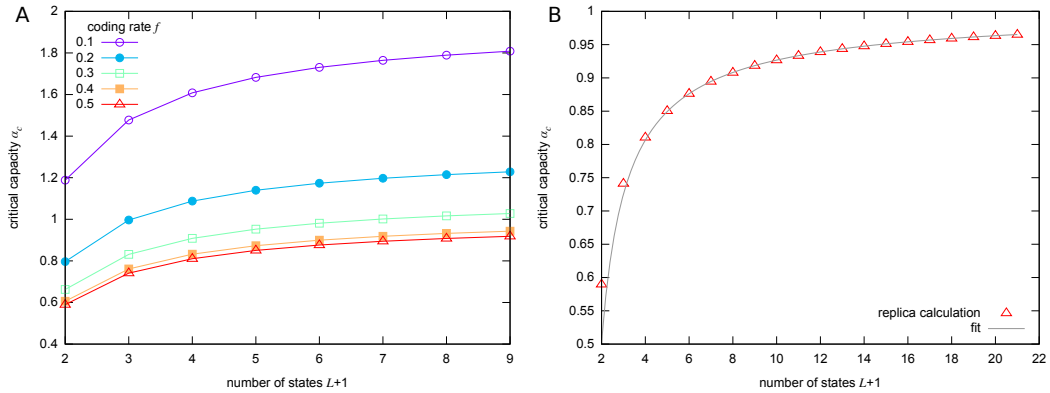

Fig. 2.1 A. Critical storage load α_c as a function of the number of available synaptic states $L+1$. The different colors refer to different coding levels f. B. Fit of the critical storage load α_c as a function of the number of available synaptic states $L+1$, when $f = 0.5$. The red triangles are points obtained through the replica calculation of the quenched entropy. The gray line is the corresponding fit.

However, the gain in storage load achievable by increasing the number of available synaptic states is significant only up to very small numbers. The critical capacity then quickly saturates to the asymptotic value $\alpha_c^{L\to\infty} = 1$. Fig. 2.1B shows an estimate of the saturation rate: we fit the theoretical result, predicted by the Gardner Analysis for $f = 0.5$, with the function:

$$\alpha_c \sim \alpha_c^{L\to\infty} - \frac{a}{L^b}, \tag{2.6}$$

with the parameters of the fit turning out to be $a \simeq 0.5$ and $b \simeq 0.85$.

This result suggests that synapses may not exploit an infinite number of synaptic states for encoding information: when a few number of bits of synaptic precision are employed, the network performances, in terms of computational or representational power, are quite soon comparable with the ones achieved in the case of infinite precision. This is in line with the belief that, the strategy implemented by biological synapses is not to infinitely increase their synaptic precision but to behave, in a very extreme case, as binary switches [40–42].

2.2.2 The geometry of the version space

The quenched entropy can only provide information on the number of optimal configurations of the synaptic couplings, as a function of the storage load. However, in order to design effective learning algorithms, it would be interesting to understand how solutions are spatially organized within the version space. In other words, we would like to set up a statistical physics analysis that aims to investigate the geometric properties of the version space.

To this end, the general strategy is to select a reference configuration of the synaptic couplings, sampled from the Boltzmann-Gibbs distribution, and then to count the number of solutions surrounding the reference configuration at a given distance.

In statistical physics, this strategy goes under the name of *Franz-Parisi* potential. It has been originally introduced in the physics of disordered systems for characterizing the nature of the meta-stable states in discontinuous mean-field spin glasses, like the p-spin spherical model. In this case, the potential plays the role of the free energy cost that has to be paid, in order to keep constant the overlap between two different configurations of the system, at different temperatures [84–86].

The quantity measuring how many solutions can be found at a given distance from a planted configuration, is known as *local entropy*:

$$S^{\text{local}}\left(\tilde{\boldsymbol{W}}, D\right) = \frac{1}{N} \log \sum_{\{\boldsymbol{W}\}} \mathbb{X}_{\xi,y}\left(\boldsymbol{W}\right) \delta\left(d\left(\boldsymbol{W}, \tilde{\boldsymbol{W}}\right) - D\right), \tag{2.7}$$

with $\tilde{\boldsymbol{W}}$ being the reference configuration of the synaptic couplings and D denoting a specific value of the distance between a solution of the learning problem and the reference configuration, computed in terms of the square distance:

$$d\left(\boldsymbol{W},\tilde{\boldsymbol{W}}\right)=\frac{1}{4N}\sum_{i=1}^{N}\left(W_i-\tilde{W}_i\right)^2, \tag{2.8}$$

where we introduce the pre-factor $1/4$ to make the measure consistent with the Hamming distance, in the extreme case of binary synaptic couplings.

The local entropy is a self-averaging quantity. Therefore, averaged over the training set, it can provide information on the *typical* number of solutions, surrounding the reference configuration:

$$S_{FP}\left(D\right)=\left\langle \sum_{\left\{\tilde{\boldsymbol{W}}\right\}} P\left(\tilde{\boldsymbol{W}}\right) S^{\text{local}}\left(\tilde{\boldsymbol{W}},D\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu},y^{\mu}\right)}, \tag{2.9}$$

where the subscript FP is a short for Franz-Parisi and $P\left(x\right)$ is the Boltzmann-Gibbs distribution. In this work, we have only considered the zero-temperature limit. As discussed, in this limit, the Boltzmann Gibbs distribution simply reduces to a flat distribution, focused on the configurations of the synaptic couplings that minimize the training error:

$$P_F\left(\tilde{\boldsymbol{W}}\right)=\frac{\mathbb{X}_{\xi,y}\left(\tilde{\boldsymbol{W}}\right)}{\sum_{\left\{\tilde{\boldsymbol{W}}'\right\}}\mathbb{X}_{\xi,y}\left(\tilde{\boldsymbol{W}}'\right)}. \tag{2.10}$$

The picture emerging from the computation of the quenched local entropy in Eq. (2.9) by means of the replica approach, does not differ from the one already depicted in the specific case of binary synaptic couplings [56], and it is robust to the tuning of both the coding level and the number of available synaptic states. Indeed, having a look at Fig. 2.2, we can clearly notice that there exists a minimum value of the distance at which the quenched local entropy (blue lines) vanishes, becoming then negative for smaller distances than the minimum one.

A negative entropy is not a physical result when dealing with discrete synaptic couplings, since, in this case, the entropy precisely counts a number of solutions.

As seen, this result should be interpreted as a failure of the replica symmetry (RS) assumption: breaking the symmetry among replicas should provide a vanishing entropy at all distances smaller than the minimum one.

The existence of a minimum distance below which no more solutions surround a reference one, as pointed out by a vanishing entropy, proves that typical solutions are actually isolated, even when generalizing the Perceptron model to multi-valued synaptic states and biased training sets.

2.3 The Large Deviation Analysis

The statistical physics analysis of the typical learning scenario in discrete Perceptron models has pointed out that, even in the generalization to multi-valued synaptic states and biased training sets, typical solutions are isolated and therefore hardly accessible by classical local-searching algorithms.

Nevertheless, as seen, there exists a set of recently proposed learning protocols, that manage to detect a solution in sub-exponential times, reaching critical capacities really close to the theoretical bound. This apparent contradiction can be solved by justifying the effectiveness of the above mentioned learning protocols, in light of the existence of regions of the version space where solutions are not isolated but organized in extremely dense clusters [78]. These sets of solutions are quite atypical for the broad range of classical local-searching algorithms, while they are extremely appealing for the newly proposed learning protocols.

In the following, we set up a large deviation analysis with the purpose of demonstrating the existence of sub-dominant but extremely dense clusters of solutions in discrete Perceptron models. To this end, we introduce an out of equilibrium measure, aiming to scale down the Gibbs weight of typical solutions, in favour of rare but extremely dense regions of solutions. The new measure then assigns an higher weight to all those configurations of the synaptic couplings surrounded by an exponential number of solutions, by re-weighting each configuration in light of the local entropy:

$$P_{RC}\left(\tilde{\boldsymbol{W}};\zeta,D\right)=\frac{\mathbb{X}_{\xi,y}\left(\tilde{\boldsymbol{W}}\right)\exp\left(\zeta NS^{\text{local}}\left(\tilde{\boldsymbol{W}},D\right)\right)}{\sum_{\left\{\tilde{\boldsymbol{W}}'\right\}}\mathbb{X}_{\xi,y}\left(\tilde{\boldsymbol{W}}'\right)\exp\left(\zeta NS^{\text{local}}\left(\tilde{\boldsymbol{W}}',D\right)\right)},\tag{2.11}$$

in the case where the reference configuration is required to be a solutions of the classification problem (the subscript *RS* stays for Re-weighted, Constrained), and:

$$P_{RU}\left(\tilde{\boldsymbol{W}};y,D\right)=\frac{\exp\left(\zeta NS^{\text{local}}\left(\tilde{\boldsymbol{W}},D\right)\right)}{\sum_{\left\{\tilde{\boldsymbol{W}}'\right\}}\exp\left(\zeta NS^{\text{local}}\left(\tilde{\boldsymbol{W}}',D\right)\right)},\tag{2.12}$$

in the case where the reference configuration is not required to be a solution of the classification problem (the subscript *RU* stays for Re-weighted, Unconstrained). The ζ parameter can be here interpreted as an inverse temperature: in the limit of $\zeta\to\infty$, the configurations with the highest statistical weight are the ones surrounded by the highest number of solutions. This out of equilibrium measure goes under the name of *Robust Ensemble* (RE).

To shed lights on the typical learning scenario described by the robust ensemble, we define the quenched free entropy density for both the constrained and the unconstrained case:

$$\Phi_{RC}\left(D,\zeta\right)=\frac{1}{N}\left\langle\log\sum_{\{\tilde{\boldsymbol{W}}\}}\mathbb{X}_{\xi,y}\left(\tilde{\boldsymbol{W}}\right)\exp\left(\zeta NS^{\text{local}}\left(\tilde{\boldsymbol{W}},D\right)\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu},y^{\mu}\right)}$$

$$\Phi_{RU}\left(D,\zeta\right)=\frac{1}{N}\left\langle\log\sum_{\{\tilde{\boldsymbol{W}}\}}\exp\left(\zeta NS^{\text{local}}\left(\tilde{\boldsymbol{W}},D\right)\right)\right\rangle_{P\left(\boldsymbol{\xi}^{\mu},y^{\mu}\right)},$$

where, in the latter definition, we have simply neglect the constraint on the reference configuration to correctly classify all the input patterns at once. Through the quenched free entropy we can compute two main observables:

- the analogue of the quenched local entropy of Eq. (2.9), counting the number of solutions surrounding a reference configuration:

$$\begin{aligned} S_{RC}(D) &= \frac{\partial}{\partial \zeta} \Phi_{RC}(D,\zeta) \\ &= \left\langle \sum_{\{\tilde{\boldsymbol{W}}\}} P_{RC}\left(\tilde{\boldsymbol{W}};\zeta,D\right) S^{\text{local}}\left(\tilde{\boldsymbol{W}},D\right) \right\rangle_{P\left(\boldsymbol{\xi}^{\mu},y^{\mu}\right)} \end{aligned} \tag{2.13}$$

- the external quenched entropy, counting the number of reference configurations embedded in a region dense of solutions:

$$\Sigma_{RC}(D,\zeta) = \Phi_{RC}(D,\zeta) - \zeta S_{RC}(D,\zeta) \tag{2.14}$$

The same holds for the unconstrained case. The constrained and the unconstrained case are strictly related: in the limit of $\zeta \to \infty$, the unconstrained case simply reduces to the constrained one. Indeed, when the reference configuration is surrounded by an extremely high number of other solutions, it is very unlikely for the reference configuration not to be a solution itself.

The constrained and the unconstrained cases are both fascinating to study for several reasons. The constrained case is directly comparable with the analysis of the typical case, described in the previous section. Instead, the unconstrained case naturally guides towards the design of new efficient learning algorithms, based on the concept of local entropy, as we will see in the next section.

In both the two cases, the results emerging from the computation of the quenched entropy by means of the replica approach may not be fully accurate. First of all, we have performed the calculation under the RS assumption, which, as seen, it is typically not verified at the saddle point in discrete Perceptron models [6]. In principle, we should then take into account at least one level of replica symmetry breaking. Second, we have estimated the solutions of the saddle-point equations numerically, thus inducing a finite level of accuracy in the final result.

In spite of this, the pictures depicted by both the constrained and the unconstrained cases are quite on the same line. This, together with the outcome of some

numerical experiments, make us confident on the final results. In the following two sections, we will discuss more in details both the constrained and the unconstrained cases.

2.3.1 Re-weighted Constrained case

In the robust ensemble, the parameter ζ plays the role of an inverse temperature like parameter, defining the density level of solutions within a cluster: the more ζ is high, the more dense the cluster is. As anticipated, we want to detect those clusters of solutions characterized by the highest density level, which corresponds to the limit of $\zeta \to \infty$.

Unfortunately, in this limit, the computation of the external quenched entropy leads to unreliable results: this quantity turns out to be negative for every choice of the number of available synaptic states and of the coding level, clearly pointing out that at least one level of symmetry breaking should be taken into account.

The need of breaking the symmetry can be explained in geometrical terms: the RS ansatz is simply assuming that there exists only one single reference configuration of the synaptic couplings optimizing the local entropy. Indeed, in the RS assumption, the overlap between two reference configurations:

$$q^{ab} = \frac{1}{N}\sum_i \tilde{W}_i^a \tilde{W}_i^b, \tag{2.15}$$

is assumed to coincide with the self-overlap:

$$q = \frac{1}{N}\sum_i \tilde{W}_i^2. \tag{2.16}$$

The failure of the RS assumption then suggests that, in the limit of $\zeta \to \infty$, the version space fragments in several highly dense clusters, such that more than one single reference configuration is surrounded by an extremely dense number of solutions.

In order to properly describe this scenario, we should then perform at least one step of replica symmetry breaking. However, already at the 1-RSB level, the saddle point equations are really hard to solve, both because there is a greater number of

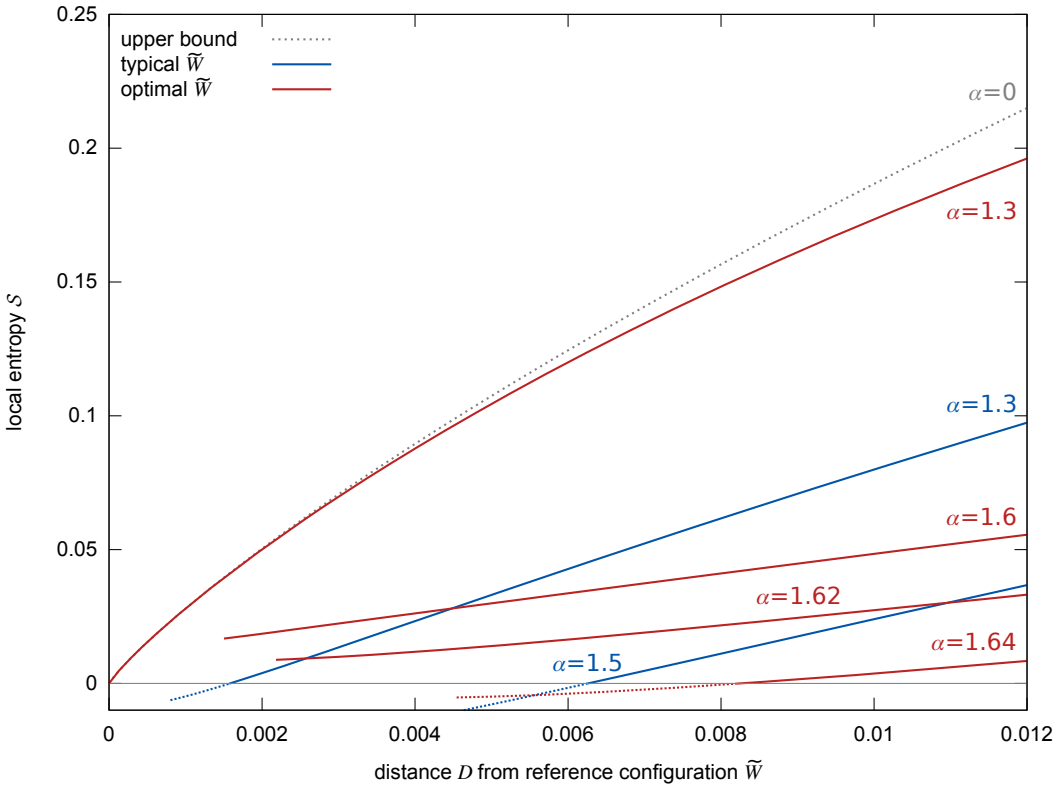

Fig. 2.2 Quenched local entropy as a function of the distance D from the reference configuration $\tilde{\boldsymbol{W}}$. Blue lines. Quenched local entropy computed through the equilibrium measure, for different values of α. Red lines. Quenched local entropy computed through the re-weighted constrained and unconstrained measures, for different values of α. In the plot $f = 0.1$ and $L = 4$.

parameters and also because some of the saddle-point equations are characterized by a series of nested integrals, whose treatment requires extremely high computational times. Moreover, even if in the limit of $\zeta \to \infty$ these technical difficulties become less challenging, it turns out that even the 1-RSB assumption is not sufficient for preventing the external quenched entropy to be negative. Then, further levels of replica symmetry breaking should actually be taken into account.

Because of that, instead of considering the limit of $\zeta \to \infty$, we can determine the maximum value of ζ for which the RS external quenched entropy is still guarantee to be non negative, namely $\zeta^\star$, and then analyse the subsequent results. Indeed, the external quenched entropy is a decreasing function of ζ: as soon as we increase ζ, it starts decreasing, till it vanishes in correspondence of the maximum value of ζ for which the RS assumption is still verified.

2.3.2 Re-weighted Unconstrained case

At first glance, the unconstrained case may seem easier to treat analytically, by looking at the definition of the corresponding quenched free entropy. Unfortunately, this is not the case: the need of breaking the symmetry among replicas is here even much stronger. In this case, even looking for the maximum value of ζ for which the RS external quenched entropy is still guarantee to be non-negative, does not lead to consistent results. For instance, the RS assumption predicts non-vanishing quenched local entropies even for values of the storage load exceeding the critical capacity of the network.

Therefore, in this case, we can not avoid to break the symmetry among replicas, by relying on the 1-RSB assumption. As in the constrained case, the saddle-point equations become easier to solve in the limit of $\zeta \to \infty$. This allows for a direct comparison with the constrained case at finite ζ, described in the previous section. It turns out that both the external quenched entropy and the quenched local entropy are indistinguishable in the two cases. This implies that the strategy we exploited in the constrained case, reasonably approximates the scenario at higher values of the temperature-like parameter ζ.

The 1-RSB assumption still predicts a negative external quenched entropy, as we have already observed in the constrained case. However, contrary to the constrained case, its modulus is now really close to zero when approaching very small distances. Moreover, the not physical solutions, predicted by the RS assumption for values of the storage load exceeding the critical capacity, no longer exist.

The emerging picture is comparable with the one already described in the extreme case of binary synaptic couplings. In particular, Fig 2.2 shows the quenched local entropy as a function of the distance from a reference configuration, sampled according to the flat measure (blue lines) in Eq. (2.10) and to the constrained and unconstrained re-weighted measure (red lines) of Eq. (2.11) and Eq. (2.12). At the resolution scale of the plot, the latter two are actually indistinguishable, then, they will be treated as a unique case in the following discussion.

For what concerns the flat measure, we can clearly notice that there exists a minimum value of the distance, below which no more solutions surrounding the reference configuration can be found. In this case, typical solutions are thus isolated.

On the contrary, for what concerns the re-weighted measure, the resulting picture is completely different:

- For storage loads smaller than a specific value α_U, the quenched local entropy converges at small distances towards the curve estimated for $\alpha = 0$, namely the limiting case where all the configurations of the synaptic couplings are solutions of the learning problem, given that no input patterns have been yet presented to the network. In this case, the reference configuration is thus surrounded by exponentially many solutions, constituting an extremely dense cluster. In the plot: $1.55 < \alpha_U < 1.62$.
- For storage loads higher than α_U but smaller than the critical capacity α_c, two different things can happen: no solutions can be found or if some solution is detected, the corresponding quenched local entropy assumes a negative value. Once again, this behaviour can be explained in terms of the geometric properties of the version space: we believe that, within this range, the dense clusters either divide in smaller components that are isolated and disconnected, or, completely disappear.

2.3.3 The α_U storage load

The α_U storage load signals the transition point where the highly dense cluster of solutions either disappears or breaks into several isolated and disconnected components. Estimating the value of α_U is not straightforward for several reasons, related to both the technical difficulties in solving numerically the saddle-point equations and to hardware issues concerning machine precision. These issues are further amplified close to the point at which the transition occurs.

Fortunately, the RS assumption provides a reasonably estimate of α_U, close to the one predicted by the 1-RSB corrections and regardless of the fact that the RS assumption can lead to some not physical results. For instance, the RS assumption estimates $\alpha_U \in (1.55, 1.62)$ for a discrete Perceptron model, with a coding level $f = 0.1$ and a number of available synaptic states $L = 4$. The 1-RSB assumption predicts instead $\alpha_U \simeq 1.6$.

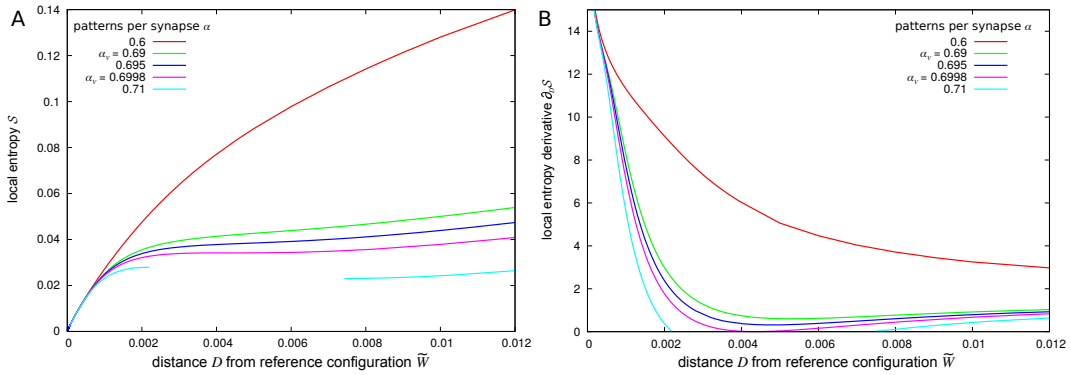

Fig. 2.3 A. Quenched local entropy as a function of the distance D from the reference configuration $\tilde{\boldsymbol{W}}$, for different values of α. We can clearly distinguish the three different phases, described in the main text. B. Same as panel A, where instead of showing the quenched local entropy, we plot its derivative with respect to the distance, as a function of the distance itself. For the network model we are here considering, that is $f = 0.5$ and $L = 2$, $\alpha_U \simeq 0.6998$.

In Fig.2.3A, we show the quenched local entropy as a function of the distance, for different values of the storage load. We can easily distinguish three different phases by tuning the network capacity:

- $\alpha < \alpha_V$: the quenched local entropy is a concave function of the distance;
- $\alpha_V < \alpha < \alpha_U$: the quenched local entropy changes its concavity in a certain range of distances, becoming a convex function;
- $\alpha_U < \alpha < \alpha_c$: the quenched local entropy breaks in two distinct branches, giving rise to a gap.

We can thus identify the transition point as the point where the gap starts appearing. Having a look at Fig.2.3B, the value of the storage load at the transition point can be then determined as the one at which the derivative of the quenched local entropy, with respect to the distance, reaches a minimum in zero:

$$\frac{\partial}{\partial D} S_{\alpha_U}\left(D_{\text{zero}}, \infty\right) = 0. \tag{2.17}$$

This behaviour is extremely robust when tuning the parameters of the model and when applying the corrections due to replica symmetry breaking. This strategy thus provides a way for estimating the value of the storage load at the transition point. To

this end, Fig. 2.4A shows the estimated values of α_U as a function of the number of available synaptic states $L+1$, for different sparsity levels f. The resulting plot is qualitatively similar to the one already provided in Fig. 2.2 and concerning the analysis of the typical learning scenario, described by the zero temperature limit of the Boltzmann-Gibbs measure. In the limit of continuous synaptic couplings, namely $L \to \infty$, both α_U and α_c are expected to collapse on the critical capacity of the continuous model, as it seems to be from the plot in Fig. 2.4B.

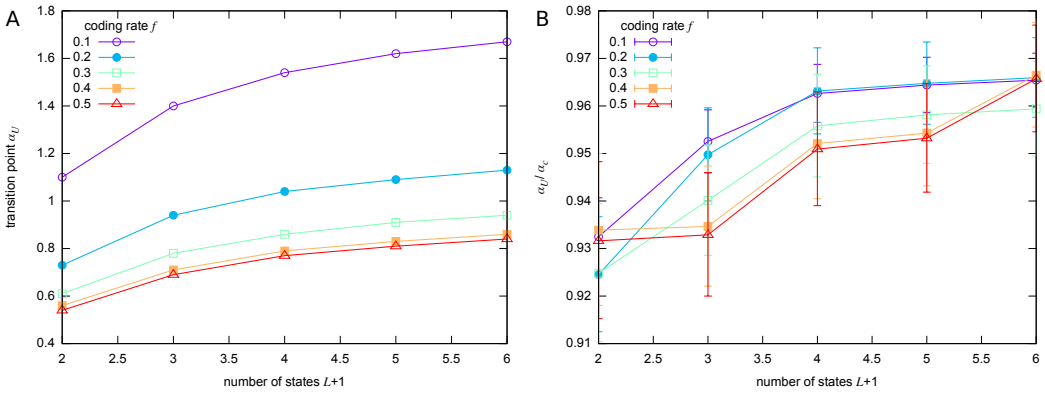

Fig. 2.4 A. α_U as a function of the number of available synaptic states $L+1$, for different values of f. B. The ratio α_U/α_c as a function of the number of available synaptic states $L+1$, for different coding levels f.

It is not yet clear if the appearance of a gap at the transition point is just a consequence of the replica symmetry assumption or if, instead, it describes a real scenario. To solve this issue, in principle, we should be able to numerically access the regions of the version space where the highly dense clusters are supposed to break in several disconnected components. Unfortunately, the recently proposed learning algorithms for the discrete learning problem, are able to detect solutions only up to values of the storage load that are really close to α_U, thus being unable to explore the learning scenario emerging at higher network capacities.

This points out a very interesting point: the efficiency of these solvers is strongly linked to the accessibility of the solutions constituting a cluster. For instance, in the discrete Perceptron model with unbiased inputs ($f = 0.5$) and two available synaptic states ($L = 1$), the predicted critical capacity is $\alpha_c = 0.59$ [43]. A stochastic version of the BPI solver (SBPI) reaches the algorithmic critical capacity $\alpha_c^{SBPI} \simeq 0.5$ [59]. Instead, the large deviation analysis predicts the value of the storage load at the transition point as $\alpha_U \simeq 0.54$, which is really close to the maximum achievable

capacity by means of the SBPI learning algorithm. The recently proposed learning algorithms are then reasonably expected to converge towards solutions belonging to the dense clusters.

The transition point underlines a change in the nature of the learning problem. Indeed, at the transition point, the learning problem switches from a phase of easily accessible solutions to a phase where they are really hard to detect. This is a quite common picture in constraint satisfaction problems, such as random K-satisfiability (K-SAT). In this case, for instance, already at the level of the equilibrium analysis, we can distinguish between an easy and an hard phase [87].

The reason why this is not the case in neural networks should be once again addressed to the geometric properties of the version space. To shed light on these scenarios, more powerful techniques, both numerical and analytic, should be designed. Up to this point, what we can say on the geometric properties of the version space in discrete Perceptron models is that the version space is extensive, there are no jumps in the density of solutions and solutions are organized in more than one, even if still less than an exponential number, extremely dense clusters.

2.4 Entropy Driven Monte Carlo

While investigating the geometry of the version space, the large deviation analysis provides a theoretical explanation of why there exist efficient algorithms able to solve the learning problem in discrete Perceptron models. As seen, these algorithms converge towards regions of the version space, where solutions are organized in extremely dense clusters. These clusters are *rare*, because sub-dominant in the typical learning scenario described by the Boltzmann-Gibbs measure, they are *accessible* to the recently proposed learning algorithms and finally, they are *robust*, since they represents an agglomeration of solutions, where each one of them is surrounded by exponentially many other solutions [88].

The large deviation analysis does not only represent a tool for shedding lights about the functioning of already existing learning algorithms. It can also be employed to design new solvers looking for highly dense regions of solutions, on the same line of how a broad range of classical algorithms exploits the equilibrium measure. In this section, we thus describe a new learning algorithm based on the concept of local

entropy, which goes under the name of *Entropy Driven Monte Carlo* (EDMC). The equivalent algorithm in the context of binary synaptic couplings is described in Ref. [89]. The proposal of another solver can instead be found in Ref. [88].

In the same vein of Simulated Annealing (SA), EDMC implements a local search, with the aim of detecting those regions of the version space, optimizing the local entropy rather than the energy of the system, as in the case of SA strategies. EDMC is based on four fundamental steps: it starts by choosing an initial configuration of the synaptic couplings, it selects one of the synaptic couplings at random, it then proposes to increase or decrease its value by one, computing the local entropy difference between the old and the newly proposed value of the synaptic coupling:

$$\Delta S^{\text{local}} = S^{\text{local}}\left(\tilde{\boldsymbol{W}}', D\right) - S^{\text{local}}\left(\tilde{\boldsymbol{W}}, D\right), \tag{2.18}$$

finally, it decides whether to accept or reject the move, according to the usual Metropolis-Hastings rule [90] at the temperature $T = \zeta^{-1}$.

After that a series of moves has been accepted, the algorithm starts increasing ζ and decreasing D, in order to look for regions of the version space, characterized by increasingly higher local entropies at smaller distances. We call *annealing* the strategy of progressively increasing ζ, in analogy with the cooling strategy of simulated annealing, and *scoping* the one of progressively decreasing D. To estimate the local entropy difference, we take advantage of the Belief Propagation algorithm. The resulting BP equations are derived in details in Appendix A.

The local entropy counts the number of solutions surrounding a reference configuration of the synaptic couplings at a given distance. As shown in Eq. (2.7), this is equivalent to impose an hard constraint on the distance between the reference configuration and a given solution, when summing over all possible configurations of the synaptic couplings. Alternatively, we can define a *local free entropy density*:

$$S_f^{\text{local}}\left(\tilde{\boldsymbol{W}}, \gamma\right) = \frac{1}{N} \log \sum_{\{\boldsymbol{W}\}} \mathbb{X}_{\boldsymbol{\xi}^\mu, \sigma^\mu}\left(\boldsymbol{W}\right) e^{\gamma d\left(\boldsymbol{W}, \tilde{\boldsymbol{W}}\right)}, \tag{2.19}$$

where the control on the distance is realized by imposing a soft constraint through the parameter γ: as γ increases, we focus on regions of the phase space closer and closer to the reference configuration. The local free entropy and the local entropy are related through a Legendre transform: instead of controlling the distance through

the parameter D, we rely on the Legendre conjugate parameter γ. The twos are linked by a bijective relation that holds for a wide range of the parameters of the model, meaning that, the hard and the soft constraint are basically interchangeable. Unfortunately, this is no more true at high values of the storage load [89].

The control parameter D is better suited for analytic and theoretical analysis. On the other hand, the parameter γ is more suitable from an algorithmic point of view: the soft constraint on the distance can be easily implemented in BP by introducing a set of external fields of intensity γ, that progressively push the configuration of the synaptic couplings towards the reference one. In this case, the scoping procedure is realized by gradually increasing γ.

Relying on a Monte Carlo strategy, EDMC is not as fast as the other learning protocols, where a solution can be achieved in less than an exponential time. Despite that, it is anyway attractive and fascinating from several points of view. First of all, it is general, in the sense that it can be applied to whatever model for which a local entropy can be defined and then computed. Second, its functioning is fully understandable from a theoretical point of view: it does not rely on heuristics for which the connection with the theoretical background is not clear at all. Finally, the last, and more important reason, is that it provides a prove on the fact that the entropy landscape is consistently different from the energy one [88].

Indeed, the numerical experiments we performed by running EDMC in comparison with SA, have shown that while SA gets trapped in one of the exponentially many local minima of the energy landscape, this is not the case for the EDMC: it can always find a solution, even when directly starting at very low temperatures, without resorting to any cooling process.

Fig. 2.5 shows an example of the energy landscape (gray curve) overlaid to the entropy landscape (red curves), for different choices of the γ parameter. The energy landscape can be very jagged: solutions are typically isolated (right minima in the plot) and embedded in a landscape rich of meta-stable synaptic states. However, the large deviation analysis has revealed the existence of extremely dense agglomerate of solutions, that give rise to wide-ranging global minima in the energy landscape (left of the plot).

The entropy landscape is instead very smooth for low values of γ (red curves). Increasing γ, the global minimum of the entropy landscape focuses more and more on the wide-ranging global minima of the energy landscape. Notice that, in the limit

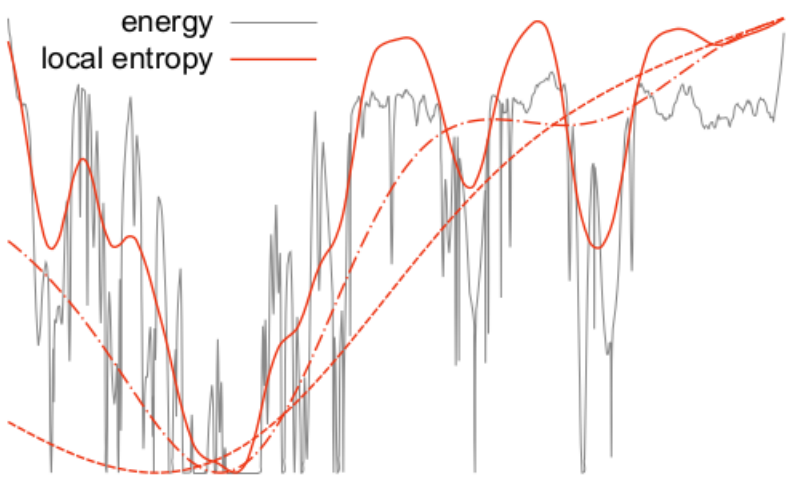

Fig. 2.5 Sketch of the comparison between the energy landscape (gray curve) and the entropy landscape for increasing value of the γ parameter (red curves) [88].

of $\gamma \to \infty$, the entropy landscape exactly matches the energy landscape. However, as soon as γ is kept finite, the smoothness of the entropy landscape allows the learning algorithms implementing a local searching, like EDMC, to easily reach the wide-ranging global minima [88].

The performance of the EDMC algorithm can be further improved with few tricks. For instance, instead of starting from a random reference configuration, we can initialize the configuration of the synaptic couplings by clipping the corresponding marginals. This can be done by running BP in absence of any constraint on the distance, till a fixed point is reached. We can also exploit BP itself for determining the move to propose instead of making a random choice.

In Fig. 2.6 we show the result of a numerical test we performed on the discrete Perceptron model with $N = 501$, $\alpha = 1.2$, $f = 0.1$ and $L = 4$ at $\zeta = \infty$. We plot the training error in log-log scale as a function of the number of Monte Carlo iterations, for four different algorithms: SA, standard EDMC, EDMC with clipped initial configurations and EDMC with the move proposal implemented through BP.

We can suddenly notice that, the number of iterations that SA requires in order to solve the classification problem, is a way longer than the number needed by the EDMC solvers. Indeed, as suggested by the plateau, SA gets stuck for a long time in a local minimum and then only eventually reaches an optimal solution. This is a characteristic feature of simulated annealing strategies when applied to glassy

systems [55, 56, 88]. In this case, the time need for escaping from a local minimum grows exponentially with the system size. On the contrary, the improvements we performed on standard EDMC make EDMC even faster.

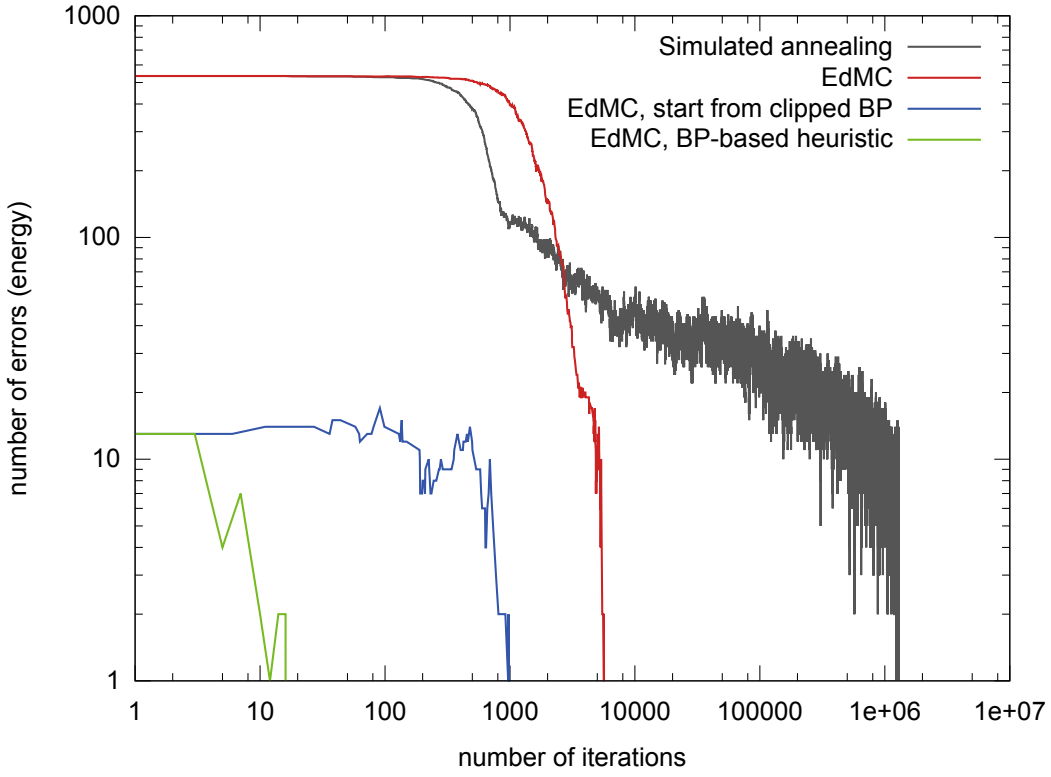

Fig. 2.6 Comparison between the SA algorithm, the standard version of the EDMC and its two variants. We plot the evolution of the training error, defined in Eq. (2.20), as a function of the number of iteration needed to reach a solution of the learning problem.

When the initial reference configuration is chosen at random, we start the scoping procedure at $\gamma = 0.5$ and then we progressively increase γ of steps of $\Delta\gamma = 1.0$. In this way, we help the convergence of the BP equations and we do not get contradictory results from BP when the reference configuration is still far to be a solution. In the case of the starting configuration of the synaptic coupling obtained by clipping the marginals, we set the starting γ at the higher value $\gamma = 3.5$.

Since SA is not able to find a solution at higher storage loads, even when implementing cooling protocols, we decided to exploit a different cost function with respect to the training error:

$$E_{\Delta}\left(\tilde{W}\right) = \sum_{\mu}\left(-\left(2y^{\mu}-1\right)\left(\sum_{i}\tilde{W}_{i}\xi_{i}^{\mu}-\theta N\right)\right)_{+}, \tag{2.20}$$

having defined $(x)_{+} = x$ if $x > 0$ and 0 otherwise. This quantity again measures the number of wrongly classified patterns. However, in this case, the correct classification of a pattern is required to be stable up to a confidence level of θN. The value of θ has to be determined through the replica calculation. The cooling scheme starts from $\zeta = 1.0$ and then proceeds by decreasing ζ at a rate of $r_{\zeta} = 1.005$, after 100 accepted moves.

Chapter 3

The Stochastic Perceptron

Feed-forward neural networks learn how to assign a set of objects to their corresponding class by building increasingly abstract representations of the input data, that enhance those traits of the object relevant for a successful classification and neglect useless details. As seen, this condition is eventually achieved at the end of a training phase, during which the network adjusts the strength of its synaptic connections, in such a way to make the response of the network coinciding with the actual class-label. By applying the soft-max operator, the response of the network is provided in terms of a vector of scores, that quantifies how likely is for a given object to belong to a specific category rather than another one [8].

Learning how to accomplish a given classification task, is thus translated into an optimization problem where, given a set of p input patterns $\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}$ and their corresponding class-labels $\{y^{\mu}\}_{\mu=1}^{p}$, the network seeks for those configurations of the synaptic couplings that optimize the probability of having correctly classified all the bunch of input patterns, namely the log-likelihood [71]:

$$\max_{\boldsymbol{W}} \mathscr{L}(\boldsymbol{W}) = \max_{\boldsymbol{W}} \sum_{\mu=1}^{p} P\left(y^{\mu} | \boldsymbol{\xi}^{\mu}, \boldsymbol{W}\right). \tag{3.1}$$

A widespread belief is indeed that learning in biological neural networks occurs at the synaptic level in terms of modulation of the synaptic efficacy [13–15]. Moreover, neural computation takes place in an extremely noisy environment [91], where it is possible to detect several sources of stochasticity, among which the unreliable release

of synaptic vesicles, induced by the pre-synaptic action potentials [92]. Synapses should then be considered as stochastic components.

It is not yet clear if stochasticity can enhance the cognitive abilities of the brain or if instead it detrimentally interferes during the learning process. Quite recently, it has been suggested that the noise acting on synaptic transmission may be essential for learning, exactly how genetic mutation is crucial for the evolution of the species [93].

In this third chapter, we thus propose a stochastic discrete Perceptron model, where synaptic couplings are treated as random variables, sampled according to a parametrized probability distribution $Q_m(\boldsymbol{W})$. In this case, learning translates into an optimization problem over the parameters of the synaptic couplings distribution:

$$\max_{\boldsymbol{m}} \mathscr{L}(\boldsymbol{m},\boldsymbol{W}) = \max_{\boldsymbol{m}} \sum_{\mu=1}^{p} \log P\left(y^{\mu}|\,\boldsymbol{\xi}^{\mu},\boldsymbol{W}\right) Q_m(\boldsymbol{W})\,. \tag{3.2}$$

The aim of this chapter is to point out how stochasticity naturally drives the optimization of the log-likelihood in Eq. (3.2) towards extremely dense clusters of solutions, which, in the second chapter, we have shown to be a characteristic trait of low precision Perceptron models.

This optimization problem presents strong analogies with *Bayesian Learning*, where the ultimate goal is to determine the whole spectrum of the synaptic couplings, each one weighted with its own probability, rather than the solely optimal configurations of the synaptic weights, as in usual learning problems [94]. In Bayesian learning, the probability distribution of the synaptic couplings is determined through the reshape of an initial guess, i.e. the *prior*, in light of the training data:

$$P\left(\boldsymbol{W}|\,y^{\mu},\boldsymbol{\xi}^{\mu}\right) \propto P\left(y^{\mu}|\,\boldsymbol{\xi}^{\mu},\boldsymbol{W}\right) P(\boldsymbol{W})\,. \tag{3.3}$$

Then, the probability of correctly classifying an input pattern not belonging to the training set, can be estimated by the Bayesian predictor, averaging over the probability distribution of the synaptic couplings:

$$P\left(y|\,\boldsymbol{\xi},\{\boldsymbol{\xi}^{\mu}\}_{\mu=1}^{p}\right) = \int d\boldsymbol{W} P(y|\,\boldsymbol{\xi},\boldsymbol{W}) P\left(\boldsymbol{W}|\,y^{\mu},\boldsymbol{\xi}^{\mu}\right), \tag{3.4}$$

where the input-output pair $(\boldsymbol{\xi}, y)$ represents the test case [94].

At first glance, optimizing the log-likelihood with respect to the parameters of the synaptic couplings distribution rather than considering the usual learning problem in Eq. (3.1), may not seem to improve the complexity of learning in discrete Perceptron models. However, we should first of all consider that the optimization can now be tackled by means of GD-based strategies [16, 18], being the parameters of the distribution assumed to be continuous variables. Second, the optimization of the log-likelihood in Eq. (3.1), typically requires the contribution of some regulirizers, like dropout [95] or the L_2-norm [73], for improving the generalization performances. In this case, instead, the fact that stochasticity leads to regions of the version space where solutions are surrounded by exponentially many other solutions, naturally improves generalization, as it has been argued by relying on Bayesian arguments: solutions located at the centre of a cluster are the ones that contribute the most to the optimal Bayesian predictor, because representative of an entire region of solutions [78, 96].

This chapter is organized in four different sections. In the first section, we describe the stochastic Perceptron model. In the second section, we propose the resulting GD-based learning algorithm. In the third section, we set up a statistical physics analysis, aiming to investigate the typical learning scenario in stochastic binary Perceptron models. Finally, in the fourth section, we further consider the specific case where the parameters of the synaptic couplings distribution are assumed to be binary variables themselves.

3.1 The Model

A *stochastic* and *binary* Perceptron is a Perceptron model whose synaptic couplings are considered as binary random variables, sampled according to the probability distribution:

$$Q_m(\boldsymbol{W}) = \prod_{i=1}^{N} \left[\frac{1}{2}(1+m_i)\delta_{W_i,1} + \frac{1}{2}(1-m_i)\delta_{W_i,-1} \right], \tag{3.5}$$

with N being the total number of neurons and $\delta_{a,b}$ being the Kronecker delta. In the following, we will denote the parameters of the probability distribution $\{m_i\}_{i=1}^{N}$

as *control parameters* or *magnetizations*, being them assumed to lie in the range $m_i \in [-1,1]$. In a Bayesian framework, this probability distribution would represent the prior of the model.

As anticipated, learning to accomplish a classification task, in binary stochastic Perceptron models, translates into an optimization problem over the control parameters of the synaptic couplings probability distribution:

$$\max_{\boldsymbol{m}} \mathscr{L}(\boldsymbol{m}, \boldsymbol{W}) = \max_{\boldsymbol{m}} \sum_{\mu=1}^{p} \log \Theta\left(y^{\mu} \sum_{i=1}^{N} \xi_i^{\mu} W_i \right) Q_m(\boldsymbol{W}), \tag{3.6}$$

where the choice about the functional form of the likelihood is the one allowing for a direct comparison with the discrete Perceptron models, discussed in the previous chapters and exploiting the training error as the cost function.

According to the central limit theorem, in the limit of $N \to \infty$, the weighted sum of the inputs is Gaussian-distributed. This allows to trace out the synaptic couplings degrees of freedom, leading to express the log-likelihood in function of the solely magnetizations:

$$\mathscr{L}(\boldsymbol{m}) = \sum_{\mu=1}^{p} \log H\left(-\frac{y^{\mu} \sum_i m_i \xi_i^{\mu}}{\sqrt{\sum_i (1-m_i^2)\left(\xi_i^{\mu}\right)^2}} \right), \tag{3.7}$$

having defined $H(x) = \int_x^{\infty} dz \exp\left(-z^2/2\right)/\sqrt{2\pi}$.

As we will see in the next section, this naturally guides towards the design of a GD-based learning algorithm for training binary Perceptrons in solving classification tasks. Indeed, when the mean squared norm of the control parameters, namely $q_* = \sum_{i=1}^{N} m_i^2/N$, tends to one, the parametrized probability distribution polarizes towards a solution of the binary learning problem:

$$Q_m(\boldsymbol{W}) \underset{q_* \to 1}{\to} \delta\left(\boldsymbol{W} - \hat{\boldsymbol{W}}\right), \tag{3.8}$$

with $\hat{\boldsymbol{W}} = \operatorname{sign}(\hat{\boldsymbol{m}})$ and $\hat{\boldsymbol{m}}$ being the configuration of the control parameter such that its mean squared norm equals one, thus optimizing the log-likelihood.

3.2 Learning in Stochastic Perceptrons

The central limit theorem allows to express the log-likelihood as a function of the solely control parameters, thus translating a binary learning problem into a continuous one. Indeed, in the stochastic Perceptron model, the control parameters are assumed to be continuous variables, lying in the range $m_i \in [-1,1]$.

Optimizing the log-likelihood with respect to the magnetizations, rather than the synaptic couplings, then makes us able to take advantage of the usual methods exploited for continuous optimization, such as the Gradient Descent algorithm and all its variants, discussed in the first chapter.

As seen, GD determines the optimal configurations of the control parameters, by updating the magnetizations along the direction of the gradient [16] of the log-likelihood:

$$\boldsymbol{m}^{t+1} \leftarrow \text{clip}\left(\boldsymbol{m}^t + \eta \frac{1}{p} \sum_{\mu=1}^{p} \nabla_{\boldsymbol{m}} \mathscr{L}(\boldsymbol{m})\right), \tag{3.9}$$

where η represents a suitable learning rate and $\text{clip}(x) = \max(-1, \min(1,x))$, to ensure the control parameters to lie in the range $m_i \in [-1,1]$. In our setting, GD has shown to be already effective in detecting solutions. In spite of this, GD variants, such as SGD [18], can be equivalently exploited. We have further tested the above proposed learning algorithm in a different scenario where, instead of optimizing the log-likelihood over the magnetizations, we have taken into account the corresponding fields $h_i = \text{arctanh}(m_i)$. This has not affected the performances, further showing a resemblance with the natural gradient heuristic [97].

The adopted learning scheme is the following. We generate the training set by sampling both the input patterns and the corresponding class labels, according to an unbiased and uniform probability distribution. We further initialize the control parameters as Gaussian random variables with zero mean and variance $1/N$. During the training, the synaptic couplings are updated according to the learning rule in Eq. (3.9). Then, at each epoch t, we check the number of wrongly classified patterns, estimating the corresponding training error:

$$\hat{E}^t = \frac{1}{p}\sum_{\mu=1}^{p}\Theta\left(y^\mu\sum_{i=1}^{N}\xi_i^\mu W_i^t\right), \tag{3.10}$$

with $\boldsymbol{W}^t = \text{sign}\,(\boldsymbol{m}^t)$. A solution to the binary learning problem is then eventually reached when the training error equals zero.

The plot on the left of Fig. 3.1 shows the GD dynamics in looking for a solution of the binary learning problem: as expected, as soon as the mean squared norm of the control parameters approaches one, the parametrized distribution of the synaptic couplings focuses on one solution of the binary learning problem, as highlighted by the descent of the training error towards zero. This behaviour presents strong analogies with the scoping strategy, described in the second chapter when dealing with the EDMC learning algorithm.

The plot on the right of Fig. 3.1 shows instead the scaling with the network size of the probability of having correctly classified all the bunch of the input patterns, as function of the storage load $\alpha = p/N$. We can clearly notice that the maximum achievable storage load is, in this case, $\alpha_{GD} \simeq 0.63$. As expected, this value is lower than the theoretical bound estimated through the Gardner Analysis $\alpha_c \simeq 0.833$ [44], but comparable with the algorithmic critical capacities reached by the recently proposed and message-passing based learning algorithms for binary Perceptrons, which lie in the range $\alpha_{MP} \in [0.6, 0.74]$ [57–60]. Moreover, as shown in the inset of the same plot, the training error is still reasonably low when exceeding the critical capacity.

3.2.1 More Biologically Plausible Variants

In order to better understand the role played by stochasticity in the process of learning, we have tried to adapt to more biologically plausible scenarios, the learning rule:

$$\boldsymbol{m}^{t+1} \leftarrow \text{clip}\left(\boldsymbol{m}^t + \eta\frac{1}{p}\sum_{\mu=1}^{p}K\left(-\frac{y^\mu\bar{h}^\mu}{\bar{\sigma}^\mu}\right)\left(\frac{y^\mu\xi_i^\mu}{\bar{\sigma}^\mu} + \frac{\left(\xi_i^\mu\right)^2\bar{h}^\mu}{\left(\bar{\sigma}^\mu\right)^3}\right)\right), \tag{3.11}$$

where we have made explicit the expression of the gradient of the log-likelihood in Eq. (3.9) and we have defined $K(x) = \partial_x \log H(x)$, $\bar{h}^\mu = \boldsymbol{m}\cdot\boldsymbol{\xi}^\mu$ and $\bar{\sigma}^\mu =$

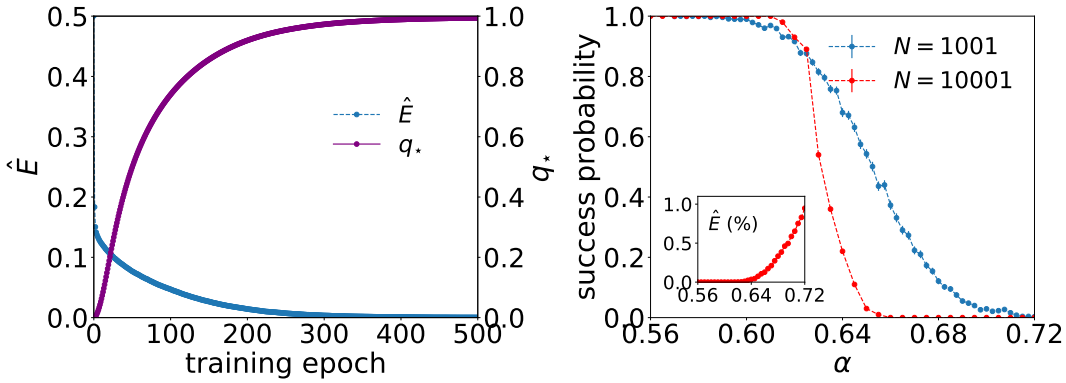

Fig. 3.1 Left. GD dynamics in looking for a solution. The training error (blue curve), namely $\hat{E}$, goes to zero as soon as the mean squared norm of the magnetizations (violet curve), namely q_*, approaches one. In the numerical simulation, $\alpha = 0.55$ and $N = 10001$. The results are averaged over 100 samples. Right. Estimation of the algorithmic capacity and its scaling with the network size. Because of finite size effects, the probability of success, namely the probability of having correctly classified all the input patters, is slightly higher for the network size $N = 10001$ (red curve), with respect to $N = 1001$ (blue curve). The results are averaged over 100 samples. In the inset. Training error as a function of the storage load, in the proximity of the algorithmic capacity.

$\sqrt{(1-\boldsymbol{m}^2)\cdot\left(\boldsymbol{\xi}^\mu\right)^2}$. We should here notice that, already at this level, the learning rule adapts the control parameters in light of the pre and post-synaptic activity, as prescribed by the Hebbian learning principle [66]. The pre-factor $K(x)$ actually plays the role of a reward signal [98].

In this model, stochasticity develops on two different levels. On one side, we should consider the stochastic nature of synaptic plasticity while, on the other side, we need to take into account that synapses are intrinsically stochastic [91, 92]. We model the first type of stochasticity by relying on SGD updates, with batch size equal to one. Whereas, we take into account the second type of stochasticity by replacing the mean pre-activation $\bar{h}^\mu$ with its single-sample realization $h^\mu = \boldsymbol{W}\cdot\boldsymbol{\xi}^\mu$, where the synaptic couplings are sampled according to $Q_m(\boldsymbol{W})$. Moreover, the mean variance $\bar{\sigma}^\mu$ is here considered as an external parameter kept fix to $\sigma = \sqrt{0.5N}$. This gives rise to a new learning rule, namely

$$\boldsymbol{m}^{t+1} \leftarrow \text{clip}\left(\boldsymbol{m}^t + \eta K\left(-\frac{y^\mu h^\mu}{\sigma}\right)\left(\frac{y^\mu \xi_i^\mu}{\sigma} + \frac{\left(\xi_i^\mu\right)^2 h^\mu}{\sigma^3}\right)\right). \tag{3.12}$$

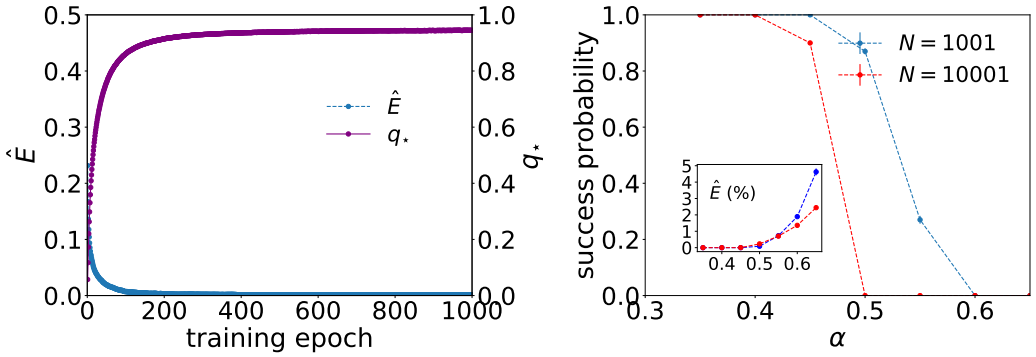

Fig. 3.2 Left. GD dynamics in looking for a solution. The training error (blue curve), namely $\hat{E}$, goes to zero as soon as the mean squared norm of the magnetizations (violet curve), namely q_*, approaches one. In the numerical simulation, $\alpha = 0.55$. Right. Probability of success, namely the probability of having correctly classified all the input patters in 2000 epochs, as a function of the storage load α and its scaling with the network size. In the inset. Training error as a function of the storage load, in the proximity of the algorithmic capacity. In both the experiments, when $N = 1001$, $\eta = 10^{-2}$ and the results are averaged over 100 samples, whereas, when $N = 10001$, $\eta = 10^{-3}$ and the results are averaged over 10 samples.

This rule acquires the structure of a Delta rule [38], already discussed in the context of spherical Perceptron models (see Eq. (1.37)). In this case, the derivative of the activation function is represented by $K(x)$.

The resulting GD dynamics is shown in Fig. (3.2). Comparing the performance of the new learning rule with respect to the ones shown in Fig. (3.1), we can clearly notice that the maximum storage load has now reduced to $\alpha \simeq 0.45$. Despite that, the rule still manages to correctly classify an extensive number of input patterns and the training error only slightly increases just above the algorithmic capacity.

Disregarding the correction in the second term of Eq. (3.12) and substituting the pre-factor $K(x)$ with an Heaviside step-function, we can obtain a further simplified version of the learning rule in Eq. (3.11):

$$\boldsymbol{m}^{t+1} \leftarrow \text{clip}\left(\boldsymbol{m}^t + \eta\left(\frac{y^\mu \xi_i^\mu}{\sigma}\right)\Theta\left(-\frac{y^\mu h^\mu}{\sigma^\mu}\right)\right). \tag{3.13}$$

This rule closely resembles the Perceptron learning rule [25], provided in Eq. (1.34), where, in this model, the role of the continuous synaptic couplings is played by the magnetizations and the mean pre-activations are replaced with their single-sample realizations. It is here important to underline that, binarizing the continuous

solutions detected by the Perceptron learning rule, does not provide a valuable strategy for solving the binary learning problem: the resulting synaptic couplings configurations are not solutions of the problem. Contrary, the stochastic version of the Perceptron learning rule in Eq. (3.13) is actually able to solve the classification problem.

The pre-activation in Eq. (3.13), namely $h^{\mu} = \boldsymbol{W} \cdot \boldsymbol{\xi}^{\mu}$, can be estimated whether choosing as synaptic couplings the sign of the magnetizations, in this case the resulting rule closely resembles the Clipped Perceptron (CP) learning rule [59, 60], or directly sampling the synaptic couplings according to $Q_m(\boldsymbol{W})$, in this case we obtain a stochastic version of the Clipped Perceptron algorithm (CP-S). However, CP very badly scales with the network size. For better scaling performances, we should rely on the learning rule in Eq. (3.12).

3.3 The Statistical Physics Analysis

In order to shed light on the effectiveness of the learning algorithm proposed in the previous section, we can set up an analysis *à la Gardner*, aiming to characterize the typical learning scenario in binary stochastic Perceptron models [30–33]. The starting point is represented, as usual, by the partition function:

$$Z = \int_{\Omega} \prod_i \mathrm{d}m_i \; \delta\left(\sum_i m_i^2 - q_* N\right) \exp\left(\beta \mathscr{L}(\boldsymbol{m})\right), \tag{3.14}$$

expressed as the integral over all magnetizations of a Gibbs-like measure, where the role of the Hamiltonian is here played by the log-likelihood in Eq. (3.7). The integration domain is defined in the range $\Omega = [-1,1]^N$ and the parameter β is the inverse temperature. We have introduced the hard constraint over the mean squared norm of the magnetizations, in order to make the theoretical analysis directly comparable with the dynamics of the gradient descent.

As specified several times, since we are interested in the typical learning scenario, a quantity for which the typical value coincides with the averaged one is represented by the quenched free entropy density. This quantity is defined as the logarithm of the partition function, averaged over the training set:

$$\Phi = \frac{1}{N} \langle \log\, Z \rangle_{P(\boldsymbol{\xi}^{\mu}, y^{\mu})} \,. \tag{3.15}$$

In order to perform the average over both the input patterns and the corresponding class-labels, we can take advantage of the usual formalism of replicas. A sketch of the calculation can be found in Ref. [99]. However, we should here to point out that, in the zero-temperature limit ($\beta \to \infty$), namely the one we are more interested in, the RS assumption does not provide stable saddle-points. This suggests that in principle, we should break the symmetry among replicas.

Instead of going through this more involved calculation, we can study the typical learning scenario in the finite temperature limit and then determine an upper bound of the temperature, beyond which the RS assumption is no longer guaranteed to be locally stable.

To this end, we can slightly perturb the RS saddle-points and look how this perturbation reflects on the quenched free entropy density. This analysis however provides a necessary but not sufficient conditions on the stability of the RS saddle-points [6]. In Fig. 3.3, we show the result of the perturbation analysis: each one of the two curves defines, for different values of α, the collection of points in the $q_{\star} - \beta$ plane at which the RS saddle-point switches from being locally stable to unstable.

3.3.1 Energy of a binarized configuration

The dynamics of the gradient descent relaxes towards a solution of the binary learning problem when the mean squared norm of the magnetizations tends to one. This relaxations can be investigated analytically in the typical scenario, by estimating the average training error associated to a binarized configuration, i.e. $\boldsymbol{W} = \text{sign}\,(\boldsymbol{m})$, as a function of the mean squared norm of the control parameters:

$$E\,(q_{*}) = \lim_{N \to \infty} \frac{1}{\alpha N} \mathbb{E}_{P(\boldsymbol{\xi}^{\mu}, y^{\mu})} \left[\sum_{\mu=1}^{p} \left\langle \Theta \left(-y^{\mu} \sum_{i} \text{sign}(m_i) \xi_i^{\mu} \right) \right\rangle \right], \tag{3.16}$$

where the average $\langle \cdot \rangle$ denotes the thermal average:

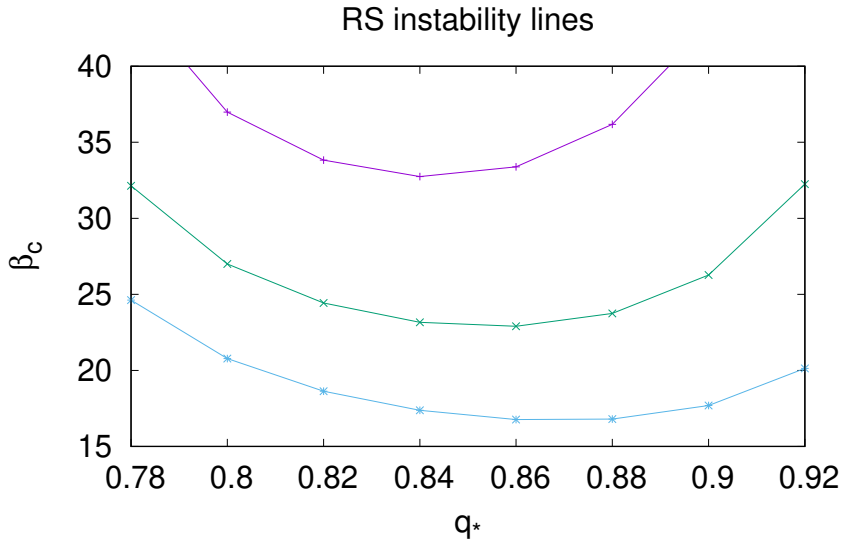

Fig. 3.3 Phase plane. For different values of the storage load, namely $\alpha = 0.5$ (purple curve), 0.55 (green curve), 0.6 (blue curve), we plot the (q_*, β_c) pairs above which the RS saddle-points are no more stable. Indeed, in our numerical simulations we set $\beta = 20$ when $\alpha = 0.55$.

$$\langle \cdot \rangle = \frac{\int \mathscr{D}\boldsymbol{m} \cdot \delta\left(\sum_i m_i^2 - q_* N\right) \exp\left(\beta \mathscr{L}(\boldsymbol{m})\right)}{\int \mathscr{D}\boldsymbol{m}\, \delta\left(\sum_i m_i^2 - q_* N\right) \exp\left(\beta \mathscr{L}(\boldsymbol{m})\right)}. \tag{3.17}$$

This quantity can be computed relying on the usual replica formalism. The plot on the left in Fig. 3.4 shows the outcome of the replica calculation (green curve), whose technical details can be found in Appendix B. This is compared with both the standard GD dynamics (blue curve) and a slow GD dynamics (orange), where the system is required to equilibrate at a fixed mean squared norm of the control parameters, thus allowing for a fairer comparison with the theoretical predictions.

The deviation of the GD dynamics from the theoretical predictions can be explained by pointing out that, the analytic result defines an equilibrium scenario, whereas the GD dynamics describes a dynamical process that can get stuck in meta-stable states. Moreover, finite size effect should be taken into account.

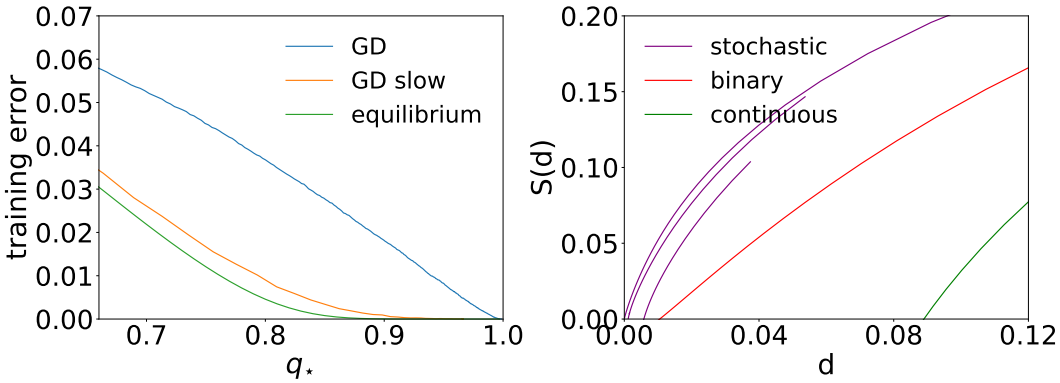

Fig. 3.4 Left. Energy of the clipped center versus the mean squared norm of the control variables q_*. Blue curve, evolution of the GD dynamics. Orange curve, GD simulation with the mean squared norm rescaled to q_* along the evolution of the gradient descent. Green curve, theoretical result, determined through the analysis of the typical learning scenario. In the simulation, $N = 10001$ and the curves are averaged over 20 samples. Right. Quenched entropy as a function of the distance D from a reference configuration. Purple curves. The reference configuration is sampled according to the thermal measure. The three curves refer to different values of the mean squared norm of the magnetizations, namely, from top to bottom, $q_* = 0.7, 0.8, 0.9$. Red curve. The reference configuration is sampled according to the uniform measure of the binary Perceptron. Green curve. The reference configuration is sampled according to the uniform measure of the spherical Perceptron and then binarized. In the simulations, $\beta = 20$, in the case of the stochastic Perceptron and $\beta = \infty$, in the case of both spherical and binary Perceptrons. In the two plots $\alpha = 0.55$.

3.3.2 The Geometry of the Version Space

In the second chapter, we have seen that typical solutions in binary Perceptrons models are isolated and embedded in a landscape rich of meta-stable synaptic states. A large deviation analysis has however revealed the existence of regions in the version space, where solutions are organized in extremely dense clusters. These solutions are a-typical at equilibrium, where synaptic couplings are distributed according to the Gibbs measure. However, they can be detected through the definition of a new measure, enhancing the statistical weight of all those solutions surrounded by an exponential number of other solutions [78, 79].

In this section, we want to provide quantitative evidence of the fact that, in the limit of $q_* \to 1$, the version space of binary stochastic Perceptron models is characterized by the presence of dense clusters of solutions.

To this end, we can resort to the Franz-Parisi approach [84–86], where the geometric properties of the version space are investigated by estimating the number of solutions that can be found at a given distance D from a reference configuration of the synaptic couplings. This quantity is provided by the constrained partition function:

$$Z(\boldsymbol{m}, D) = \sum_{\{W_i\}} \prod_{\mu=1}^{p} \Theta\left(-y^{\mu} \sum_{i=1}^{N} W_i \xi_i\right) \delta\left(ND - \sum_{i=1}^{N} \text{sign}\,(m_i)\, W_i\right). \tag{3.18}$$

To describe the typical learning scenario, we need, as usual, to average over both the training set and the thermal measure:

$$S(D) = \lim_{N\to\infty} \frac{1}{N} \mathbb{E}_{P(\boldsymbol{\xi}^{\mu}, y^{\mu})} \langle \log Z(\boldsymbol{m}, D) \rangle \,. \tag{3.19}$$

Once again, this computation can be tackled by means of the replica formalism. A sketch of the entire calculation is provided in Ref. [99] . The plot on the right of Fig. (3.4) shows the Franz-Parisi entropy $S(D)$ (purple curves) as a function of the distance D from a reference configuration, sampled according to the thermal measure. The three purple curves refer to increasingly higher values of q_*.

We can clearly notice that, as q_* approaches one, the distance at which the Franz-Parisi entropy vanishes, tends to zero, meaning that the reference configuration is no more isolated but surrounded by an exponential number of solutions. For comparison, in the same plot, we show the same quantity when the reference configuration is sampled according to the uniform measure of the spherical Perceptron (green curve) [30] and the one of the binary Perceptron (red curve) [56]. We can thus notice that, contrary to the stochastic binary Perceptron model, the typical configurations obtained by binarizing the ones of the spherical Perceptron and the typical solutions of the binary Perceptron model are isolated.

As a proof of concept, we can perform a numerical experiment, where we select the reference configuration by running GD and rescaling the mean squared norm of the magnetizations to q_* after each update, till convergence. We switch to the next value of q_* exploiting a slow updating schedule. We finally estimate the corresponding Franz-Parisi entropy through Belief Propagation. The plot in Fig. (3.5) shows the comparison between the outcome of this numerical experiment and the result of the replica calculation.

Once again, the slightly difference between the twos can be addressed to different reasons. First of all, we need to consider that, because of the instability of the RS saddle-point at zero-temperature, we perform the replica calculation at $\beta > 0$. Second, GD describes a dynamical process through which we select the reference configuration. On the contrary, in the replica calculation, the reference configuration is selected according to the thermal measure of Eq. (3.17), describing the system at equilibrium. Finally, even in this case we should take into account finite size effects.

To shed more lights on the geometric properties of the version space in binary stochastic Perceptron models, we introduce a new observable defined as the average training error, associated to the set of synaptic configurations surrounding a reference solution of the learning problem:

$$\hat{E}\left(\boldsymbol{W}, D\right) = \mathbb{E}_{\boldsymbol{W}|\boldsymbol{W}', D} \frac{1}{p} \sum_{\mu=1}^{p} \Theta\left(-y^{\mu} \sum_{i=1}^{N} W_i \xi_i\right), \tag{3.20}$$

where we denote with $\boldsymbol{W}$ the reference solution and with $\boldsymbol{W}'$ one of the configurations of the synaptic couplings placed around it. Here, the average is defined as:

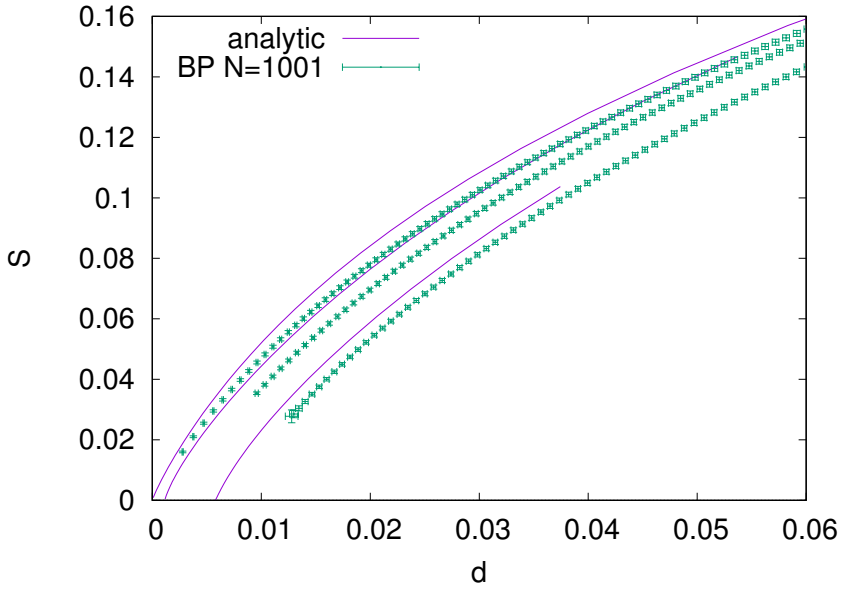

Fig. 3.5 Franz-Parisi entropy as a function of the distance D from a reference configuration at $\alpha = 0.55$ and for different values of the mean squared norm of the magnetizations, namely, in increasing order, $q_* = 0.7, 0.8, 0.9$. In the simulation, $\beta = 20$. Purple curves. Franz-Parisi entropy computed by means of the replica approach and describing the typical learning scenario. Green curves. Franz-Parisi entropy estimated by means of Belief Propagation. In both simulations, the results are averaged over 100 samples.

T	SA	CP	CP-S	GD	BP+R
1	0.578(3)	0.628(3)	0.644(3)	0.642(3)	0.657(2)

Table 3.1 Comparison on the generalization accuracy of the five different learning algorithms: simulated annealing (SA), clipped Percptron (CP), stochastic clipped Perceptron (CP-S), gradient descent (GD) and Belief Propagation plus reinforcement (BP+R), in the teacher-student learning scenario. In the simulations, $N = 1001$ and $\alpha = 0.4$.

$$\mathbb{E}_{\boldsymbol{W}|\boldsymbol{W}',D}\,\cdot = \frac{\sum_{\boldsymbol{W}'}\cdot\,\delta\left(N(1-2D)-\sum_{i=1}^{N}W_iW_i'\right)}{\sum_{\boldsymbol{W}'}\delta\left(N(1-2D)-\sum_{i=1}^{N}W_iW_i'\right)}. \tag{3.21}$$

This quantity allows to provide more insights on the nature of the synaptic configurations surrounding a reference solution, thus providing information on the heterogeneity of the energy landscape. Therefore, the configurations of the synaptic couplings $\boldsymbol{W}'$ do not necessarily represent solutions of the learning problem, they can be then easily sampled from the reference solution by randomly flipping dN spins.

To compare the performances of the learning algorithm proposed in this chapter, with the ones of others already existing learning protocols, we extract as reference solution a solution of the teacher-student learning problem according to the learning rule (GD) in Eq. (3.11), the two variants in Eq. (3.12) (CP and CP-S), the BP+R learning algorithm for binary Perceptrons and, finally, the SA algorithm with the objective function $H = \sum_{\mu=1}^{p}\left(\left(y^{\mu}/\sqrt{N}\right)\boldsymbol{W}'\cdot\boldsymbol{\xi}^{\mu}\right)\Theta\left(-\left(y^{\mu}/\sqrt{N}\right)\boldsymbol{W}'\cdot\boldsymbol{\xi}^{\mu}\right)$.

The results are shown in Fig. 3.6. We can clearly notice that, SA (green curve) experiences a sharp jump in the average training error as soon as we move from the reference solution, further approaching the limit where the reference solution is represented by the teacher itself (purple curve). Indeed, it is known that the teacher represents an isolated solution of the binary model. Contrary, the solutions identified by the other learning protocols are surrounded by configurations characterized by lower training errors.

Moreover, we can also notice that, the solutions determined by the CP learning algorithm are characterized by a less flattered energy landscape, despite the fact that they are not isolated as the ones detected by SA.

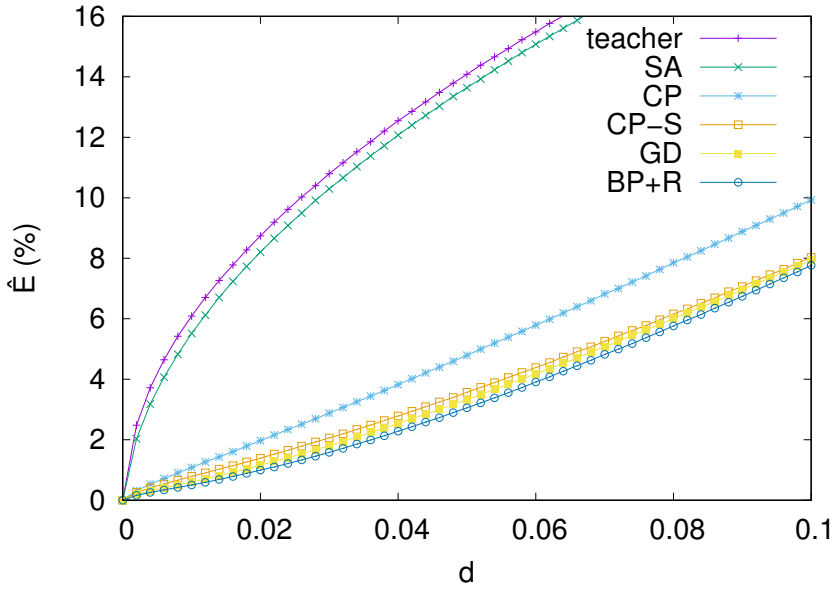

Fig. 3.6 Averaged training error as a function of the Hamming distance of the synaptic configurations from the teacher and from a given reference solution, detected by means of the five different learning algorithms. The simulations refer to $N = 1001$ and $\alpha = 0.4$ and they are averaged over 40 samples. In CP and CP-S the learning rate is set at $\eta = 2 \times 10^{-3}$, instead in standard GD it is $\eta = 0.1$. In BP+R, the reinforcement term is adjusted according to the protocol $1 - r^{t+1} = (1 - r^t)\left(1 - 10^{-3}\right)$. Finally, cooling in SA is performed according to the schedule $\beta^{t+1} = \beta^t \left(1 + 5 * 10^{-3}\right)$.

To compare the generalization performances of the above mentioned learning algorithms, we tested each one of them on the classification of a new pattern, not belonging to the training set. In Table (3.1), we show the probability of having correctly classified the new test case. We can clearly notice that, there exists a link between the smoothness of the energy landscape and the generalization performances. Indeed, SA annealing is the one showing the worst generalization ability. The technical details concerning the numerical simulations can be found in Ref. [99].

3.4 Binary control parameters

A more explicit connection with the large deviation analysis described in the second chapter, can be done by assuming the magnetizations to be binary variables, namely $m_i = \sqrt{q_*}\tilde{m}_i$ with $\tilde{m}_i = \{-1, 1\}$. The definition of the log-likelihood then needs to be modified in light of the new assumption:

$$\mathscr{L}(\tilde{\boldsymbol{m}}) = \sum_{\mu=1}^{p} \log H\left(-\rho \frac{y^{\mu}\sum_i \tilde{m}_i \xi_i^{\mu}}{\sqrt{\sum_{i=1}^{N}\left(\xi_i^{\mu}\right)^2}}\right), \tag{3.22}$$

where we have defined $\rho = \sqrt{\frac{q_*}{1-q_*}}$. The statistical physics analysis we set up in the previous section can be here performed in the same way. The estimation of the main quantities thus introduced, can be carried out by relying on the usual replica formalism.

In the specific case of binary control parameters, we can not rely on GD-based strategies in order to optimize the corresponding log-likelihood. We can instead implement a Monte Carlo method, where the energy function is represented by the log-likelihood in Eq. (3.22). In Fig. (3.7), we show the behaviour of the training error with respect to q_*. The result of the replica calculation (green curve) is quite in good agreement with the outcome of the Monte Carlo simulation.

The case of binary magnetizations is valuable for study also because, in this setting, we can identify an interesting link with the dropout/drop-connect strategy, typically exploited in deep-learning for improving the generalization performances of a neural network. In the dropout scheme, some of the inputs are randomly neglected, according to some probability p [95]. Whereas, in the drop-connect scheme, are

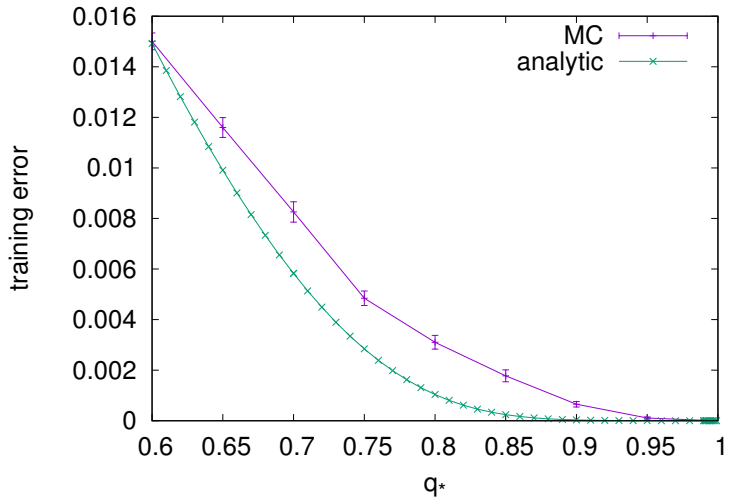

Fig. 3.7 Training error as a function of q_*. The green curve provides the result of the replica calculation, whereas the purple curve shows the outcome of the Monte Carlo simulation, performed at $N = 1001$ and averaged over 100 samples. In both cases $\alpha = 0.55$ and $\beta = 15$.

the synaptic connections to be progressively erased [100]. We can then consider an equivalent stochastic model where some of the input neurons, or some of the synaptic connections, are randomly deleted. The log-likelihood of this model simply matches with the one in Eq. (3.22), provided that it holds $p = 1 - q_*$.

Chapter 4

The Delayed Correlation Matching learning rule

Learning and memory are strongly believed to occur in biological networks in terms of modulation of the synaptic efficacy [13–15]. This phenomenon has widely inspired machine learning methods as well as the design of neuromorphic devices. It goes under the name of *synaptic plasticity* [101]. In biological networks, synaptic plasticity is usually modelled through learning principles, that tend to link the modulation of the synaptic strength to both the pre and post-synaptic activity. All these principles share a common ancestor, namely the Hebbian learning principle [66].

Hebbian learning relies on the assumption that long term synaptic potentiation (LTP) is induced by the reiterated and persistent stimulation of post-synaptic neurons by means of the pre-synaptic ones. On the contrary, long term depression (LTD) arises when the activity of the pre-synaptic neurons does not affect post-synaptic cells. The Hebbian learning principle is often conveniently synthesized with the expression "*cells that fire together are wired together*"[66].

In the first chapter, we have seen that Hebbian learning can induce an attractor dynamics in recurrent neural networks [6]. In the past years, this capability of the learning principle has generated great excitement in Neuroscience, since it provides a way of modelling the different expressions of persistent neuronal activity that have been observed within several areas of the brain [102, 103].

In this regard, the first attempt in modelling associative memory is represented by the Hopfield model [65]. As seen in the first chapter, in the Hopfield model, the minima of an Ising-like Hamiltonian can be identified with the stable attractors of the network dynamics, when the synaptic couplings are selected according to the Hebbian learning principle. The core idea of representing memories as energetically favoured states of an Ising-like Hamiltonian, has been suggested by the very well known phenomenon of *frustration* in disordered systems [24].

Despite the fact that Hebbian learning allows the storage of an extensive number of unbiased and uncorrelated patterns of neuronal activity, several variants and re-statements have been suggested over the years in order to adapt the Hebbian learning principle to wider settings. For instance, as seen in the first chapter, when considering the more realistic scenario of correlated patterns of activity, the Hebbian learning rule has to be modified by encoding in the synaptic couplings, the information concerning the statistics of the input stimulus [69]. The Hebbian learning principle has been further adapted to rates neural networks models [104] up to the more recently proposed spiking neural networks [105].

When trying to get more insights on phenomena like learning or memory retrieval, we need to consider that the neuronal environment is extremely noisy [91]. Within the brain, stochasticity takes place at different levels and scales. For instance, we could mention the variability of neuronal responses to the same external stimulus [106, 101], the untrustworthy transport of synaptic vesicles [92], the strong fluctuations in the opening and closing of the ion channels [107] as well as the ripples of the neuronal membrane potential [64]. The sensory input itself is characterized by a not negligible noisy component [108].

Therefore, it would seem reasonable to take into account stochastic models of neural networks and then to exploit the strong connection between learning and inference problems [109]. Boltzmann machines work in this direction. As seen in the first chapter, they are able to construct an internal representation of the sensory input, allowing the network to generate new data samples that, in principle, are indistinguishable from the input itself [72]. However, as already pointed out, training a Boltzmann Machine is a challenging task, mostly because relying on Monte Carlo sampling [74]. Therefore, simpler models, such as Restricted Boltzmann Machines (RBMs), have been taken into account [75].

In biological neural networks, synaptic connections are typically asymmetric and time varying [68]. This observation has motivated the development of purely kinetic models, whose ultimate goal is the analysis of the dynamics of learning processes. However, the research in this field requires great computational efforts. Indeed, if there exist stationary states of the dynamics, it is not straightforward how to deal with them both from an analytic and a numerical point of view [110–112].

In this chapter, we will try to face all these different issues, with the purpose of designing a more realistic and biological amenable learning model in recurrent neural networks. This will naturally guide us towards the proposal of a new learning rule, allowing a recurrent neural network to store a set of patterns of neuronal activities as well as to build an internal representation of the sensory world.

In the following, we will thus consider stochastic recurrent neural networks, whose synaptic connections are not symmetric and modulated according to information only locally available to each synapse. Learning will be treated as an unsupervised process, namely not influenced by strong external stimulus or feed-back signals, as in the case of learning from examples. The resulting learning rule turns out to be consistent with the Dale's principle [80, 81] and effective in an on-line inspired learning scenario. Moreover, notice that, in the course of this work, we will only consider a discrete-time dynamics, even if the model can be generalized to continuous time.

This chapter is organized as follows. In the first section, we will present the learning model driving towards the definition of the new learning rule. In the second section, we will consider the case of a recurrent neural network made of solely visible units. In this case, we will extend the new learning rule to more biologically plausible settings, by introducing some additional constraints, such as weak external signals and sparse neural coding. We will then compare the performances of the new rule with respect to Hebbian learning, when dealing with correlated patterns, on-line learning scenarios and spurious attractors. In the third section, we will instead apply the new learning rule in training Restricted Boltzmann Machines. We will test its performances on features extraction and classification tasks, comparing the results with the performances achieved by the state-of-the-art in RBMs training described in the first chapter, namely Contrastive Divergence [77]. All the technical details are provided in Appendix C.

4.1 The model

The neural network we are are going to deal with in this chapter is the one of a recurrent neural network, made of N neurons and characterized by asymmetric synaptic couplings, namely $W_{ij} \neq W_{ji}$. The state of each neuron s_i evolves in time according to the Glauber transition probability [64]:

$$P\left(\boldsymbol{s}'|\boldsymbol{s};\beta\right) = \prod_{i=1}^{N} \sigma\left(s_i'|h_i;\beta\right), \tag{4.1}$$

describing a discrete and synchronous dynamics. Here, $\sigma(x) \propto \exp(\beta h_i(x))$ is a sigmoid-shaped function and h_i is the local field, defined as the sum of two distinct contributions, namely the external field and the field exerted by the surrounding neurons on neuron i: $h_i = h_i^{ext} + \sum_{j\neq i} W_{ij} s_j - \theta_i$, with θ_i being the local threshold. Neurons are thus treated as intrinsically stochastic components, with the temperature $T = 1/\beta$ actually playing the role of noise. The dynamics described by the Glauber transition probability in Eq. (4.1) is ergodic, provided that the external field does not depend on time and that the synaptic couplings do not diverge [110]. In this case, it can relax towards a stationary state, whose analytic expression can not be derived explicitly in the context of kinetic models, namely when synaptic couplings are assumed to be asymmetric [110].

Our goal is to design a new learning rule allowing the network to store a set of p patterns of neuronal activity $\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}$ or, more generally, to build an internal representation of the statistics of the inputs $\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}$, while not relying on any additional supervising or feed-back signal.

To this end, we distinguish two different subset of neurons: the visible neurons are the ones subject to the external input, whereas the hidden neurons are the ones left free to catch the complex correlations in the statistics of the input patterns. We then consider the following learning scenario. An input pattern $\boldsymbol{\xi}^{\mu}$ is presented to the visible units as an external stimulus, under the form of an external field of intensity λ^{ext} and oriented along the direction of the input pattern itself. The intensity of the external field is chosen in order to only bias the activity of the network towards the input pattern, thus preventing the network to be completely clamped on the external stimulus.

After this initial phase, the external field starts rapidly vanishing, namely $\lambda^{ext} \to 0$. We then want both the synaptic couplings and the thresholds to adapt in order to compensate the effect of the vanishing external field. When this is the case, the dynamics of the network left free to evolve without any supervisory signal, closely resembles the one prompted by the external stimulus. At the end of the training, starting from a noisy version of the input pattern, the network is then eventually able to retrieve the original pattern, while not relying on any biasing signal.

This condition can be achieved by requiring to modulate the strength of the synaptic couplings and the thresholds in such a way to reduce the Kullback-Leibler (KL) divergence between the dynamics of the network in presence of an external field of intensity λ_1^{ext}, and the dynamics of the same network in presence of a weaker external field of intensity λ_2^{ext}:

$$\begin{aligned} &\left\langle \mathrm{KL}\left(P\left(\cdot|\boldsymbol{s};\lambda_1^{ext},\beta\right)||P\left(\cdot|\boldsymbol{s};\lambda_2^{ext},\beta\right)\right)\right\rangle_{P(\boldsymbol{s})} = \\ &= \sum_{\{\boldsymbol{s}\}} P(\boldsymbol{s}) \sum_{\{\boldsymbol{s}'\}} P\left(\boldsymbol{s}'|\boldsymbol{s};\lambda_1^{ext},\beta\right) \log \frac{P\left(\boldsymbol{s}'|\boldsymbol{s};\lambda_1^{ext},\beta\right)}{P\left(\boldsymbol{s}'|\boldsymbol{s};\lambda_2^{ext},\beta\right)}, \end{aligned} \tag{4.2}$$

where $P(\boldsymbol{s})$ denotes the average over all possible initial states of neuronal activity, which we choose in such a way to make the initial state of the network close to the pattern induced by the external stimulus. Because of that and because of the presence of both the external field and the recurrent stimulus exerted by surrounding neurons, the successive network states, namely $\boldsymbol{s}'$, will remain close to the input pattern.

For the sake of simplicity, we start considering the extreme case where the external field is initially applied with infinite intensity, namely $\lambda_1 \to \infty$, and then suddenly dropped to zero, namely $\lambda_2 = 0$. As we will point out, this learning scenario allows for a straightforward comparison with Hebbian learning. Applying an infinite external field, makes the state of the visible units to be completely clamped on the pattern of neuronal activity imposed by the external stimulus. The distribution over the initial network states then assumes the simpler factorized form:

$$P_{\mathrm{clamp}}(\boldsymbol{s};\boldsymbol{\xi}) = \left(\prod_{i\in V} \delta_{s_i,\xi_i}\right) P(\boldsymbol{s}_H|\boldsymbol{s}_V), \tag{4.3}$$

where δ_{xy} is the Kronecher Delta and V and H denote respectively the subset of visible and hidden units. For the time being, we do not write an explicit expression of the distribution over the hidden network states, this would invalidate the generality of the present discussion.

In this case, optimizing the KL divergence in Eq. (4.2), with $\lambda_1 \to \infty$ and $\lambda_2 = 0$, leads to a learning rule for both synaptic couplings and thresholds, that acquires the form of a matching condition between activity correlations, subject to different field intensities:

$$\begin{aligned} \Delta W_{ij} &\propto \left(\left\langle s'_i s_j \right\rangle_{\text{clamp},\infty} - \left\langle s'_i s_j \right\rangle_{\text{clamp},0} \right) \\ \Delta \theta_i &\propto - \left(\left\langle s'_i \right\rangle_{\text{clamp},\infty} - \left\langle s'_i \right\rangle_{\text{clamp},0} \right), \end{aligned} \tag{4.4}$$

where $\boldsymbol{s}$ and $\boldsymbol{s}'$ are two consecutive configurations of neuronal activity and $\langle \cdot \rangle_{\text{clamp},\lambda^{\text{ext}}}$ denotes the average over all possible states that can be reached starting from the initial one, namely

$$\begin{aligned} \left\langle s'_i s_j \right\rangle_{\text{clamp},\lambda^{\text{ext}}} &= \sum_{\{\boldsymbol{s}',\boldsymbol{s}\}} s'_i s_j \, P\left(s'_i | \boldsymbol{s}; \lambda^{\text{ext}}\right) P_{\text{clamp}}\left(\boldsymbol{s}; \xi\right) \\ \left\langle s'_i \right\rangle_{\text{clamp},\lambda^{\text{ext}}} &= \sum_{\{\boldsymbol{s}',\boldsymbol{s}\}} s'_i \, P\left(s'_i | \boldsymbol{s}; \lambda^{\text{ext}}\right) P_{\text{clamp}}\left(\boldsymbol{s}; \xi\right). \end{aligned} \tag{4.5}$$

The analytic derivation of the update rules in Eq. (4.4) can be found in Appendix C. Notice that, in the limit of strong external fields, the activity of the network is completely clamped on the input pattern, such that it holds $\left\langle s'_i s_j \right\rangle_{\text{clamp},\infty} = \xi_i \xi_j$. Hebbian learning thus arises as the limiting case of the above described learning scenario, when considering strong biasing signals.

In biological networks, external stimuli are known to be comparable with the recurrent signal exchanged among the neurons within the network. Therefore, with the purpose of modelling a more biologically plausible learning scenario, we need to leave the setting of strong biasing signals, in light of a more realistic framework, where the intensity of the external signal, acting on a given neuron, is close to the one exerted by surrounding cells.

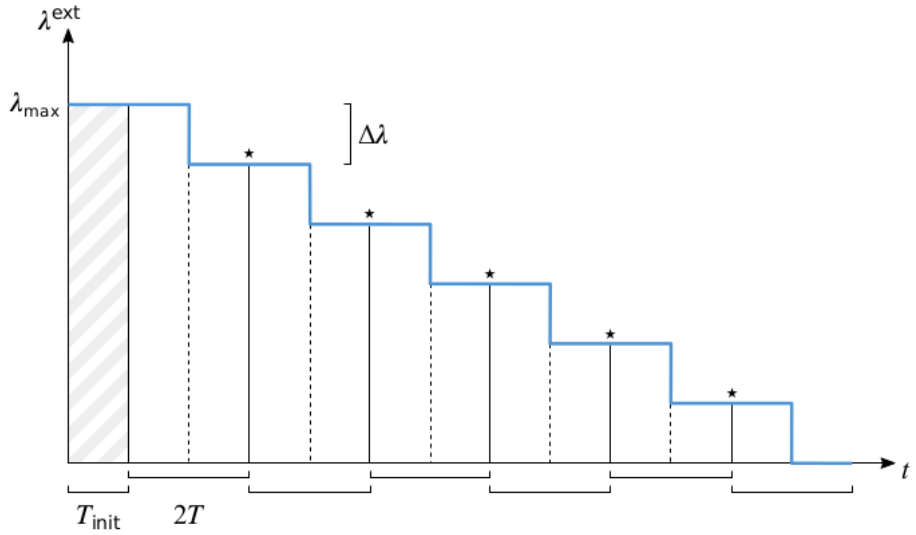

Fig. 4.1 Qualitative drawing of the descent of the external field. The symbol $\star$ highlights the time at which the update of the synaptic couplings and thresholds takes place.

In the following, we will thus consider the more realistic scenario of a finite and gradually vanishing external field. The descent of the field develops according to the following schedule. We initialize the external field to a finite value, namely $\lambda^{ext}(t=0) = \lambda_{max}$. We then start decreasing the field intensity of steps of $\Delta\lambda$ after $2T$ steps of the network dynamics, till reaching zero. The learning rule turns out to be robust to whatever choice of these external parameters. We provide a sketch of the descent of the external field in Fig. (4.1).

In this more biologically plausible scenario, the learning rule retains the form of a matching condition, where the correlations are now the ones induced by the external fields of intensities $\lambda_1 = \lambda$ and $\lambda_2 = \lambda - \Delta\lambda$, namely

$$\begin{aligned} \left\langle s_i' s_j \right\rangle_\lambda &= \sum_{\{\boldsymbol{s}',\boldsymbol{s}\}} s_i' s_j P\left(s_i'|\boldsymbol{s};\lambda\right) P(\boldsymbol{s}) \\ \left\langle s_i' \right\rangle_\lambda &= \sum_{\{\boldsymbol{s}',\boldsymbol{s}\}} s_i' P\left(s_i'|\boldsymbol{s};\lambda\right) P(\boldsymbol{s}) . \end{aligned} \tag{4.6}$$

This time, the distribution $P(\boldsymbol{s})$ just places the state of the visible neurons around the input pattern, instead of clamping it.

As it is shown in Appendix C, estimating correlations in kinetic models can be tackled by means of mean field methods, despite the lack of an Hamiltonian. In the following, we will however consider the more realistic learning scenario where correlations are directly estimated from the network dynamics. Indeed, if the dynamics of the network is ergodic, the ensemble average $\langle\cdot\rangle_{\lambda}$ can be identified with the temporal average over all the states explored by the network dynamics. This condition is achieved when the initial field intensity, namely λ_{max}, is chosen strong enough to border the dynamics of the network in the basin of attraction of the input pattern, and when the evolution of the activity of the network in time, stays close to this region as soon as the external field starts vanishing. In the basin of attraction, the network is then free to explore every possible configuration of the neuronal activity.

The resulting learning rule for synaptic couplings and thresholds, namely

$$\begin{aligned} \Delta W_{ij} &\propto \left(\left\langle s_i^{t+1} s_j^t \right\rangle_{t,\lambda} - \left\langle s_i^{t+1} s_j^t \right\rangle_{t,\lambda-\Delta\lambda} \right) \\ \Delta\theta_i &\propto -\left(\left\langle s_i^{t+1} \right\rangle_{t,\lambda} - \left\langle s_i^{t+1} \right\rangle_{t,\lambda-\Delta\lambda} \right), \end{aligned} \tag{4.7}$$

goes under the name of *Delayed Correlation Matching* (DCM). A pseudo-code of the learning algorithm is provided in Appendix C.

4.2 Fully-Visible

The learning model described in the previous section, is relative to the general case of recurrent neural networks constituted by both visible and hidden units and characterized by asymmetric synaptic couplings. In this section, we instead focus on the specific architecture of a recurrent neural network made of solely visible neurons. As said, in the case of symmetric synaptic couplings, the network dynamics is characterized by the presence of specific patterns of neuronal activity, playing the role of attractors for the network dynamics. However, kinetically persistent neuronal states can be identified even in the case of asymmetric synaptic couplings, thus preserving the concept of attractor state.

In this case, the DCM learning rule is required to make the network able to store a set of $p = \alpha N$ input patterns $\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p}$, with $\xi_i^{\mu} \in \{-1, 1\}$ and α being the storage load.

In the context of strong biasing external signals suddenly vanishing to zero, the optimization of the KL divergence translates into an on-line version of a log-likelihood factorized approximation, known as *log-pseudo-likelihood* [71, 113]:

$$\begin{aligned} &\mathscr{L}\left(\left\{\boldsymbol{\xi}^{\mu}\right\}_{\mu=1}^{p} | W_{ij}, \theta; \beta\right) = \\ &= \frac{1}{p} \sum_{\mu=1}^{p} \sum_{i=1}^{N} \log P\left(s_i = \xi_i^{\mu} | \left\{s_j = \xi_j^{\mu}\right\}_{j \neq i}; \lambda^{ext} = 0\right). \end{aligned} \tag{4.8}$$

The resulting DCM plasticity rule, namely

$$\Delta W_{ij} \propto \left(\xi_i^{\mu} \xi_j^{\mu} - \sum_{s_i^{t+1}} P\left(s_i^{t+1} | \left\{s_j = \xi_j^{\mu}\right\}_{j \neq i}; \lambda^{ext} = 0\right) s_i^{t+1} \xi_j^{\mu}\right), \tag{4.9}$$

can then be derived through the optimization of the log-pseudo-likelihood in Eq. (4.8) or, equivalently, through the one of the KL divergence in Eq. (4.2).

In this learning scenario, DCM shows two features that make it to significantly differ from Hebbian learning. First of all, it preserves the asymmetry of the synaptic couplings. Second, it relies on the difference between two distinct contributions, namely the Hebbian contribution and a second term, known as *comparator*. The presence of the comparator prevents the appearance of possible out-of-control positive feed-back loops, by avoiding to update the synaptic couplings in case of matching between the two contributions. Indeed, if the information carried by the external signal has already been encoded within the synaptic couplings, there is no reason for them to adapt. We can further notice that, in the limit of $\beta \to \infty$, the DCM learning rule translates into the Perceptron learning rule in Eq. (1.34).

Fig. (4.2) shows the maximum achievable storage load as a function of the required size of the basin of attraction, for both the DCM learning rule (blue curve)

and Hebbian learning (red curve). We can clearly notice that, the critical capacity of the DCM plasticity rule is always well above the one reached by Hebbian learning, namely $\alpha_c = 0.138$, despite the fact that it still stays below the limit of $\alpha_c = 2$, set by the Gardner Analysis, because of the sources of stochasticity introduced in the model.

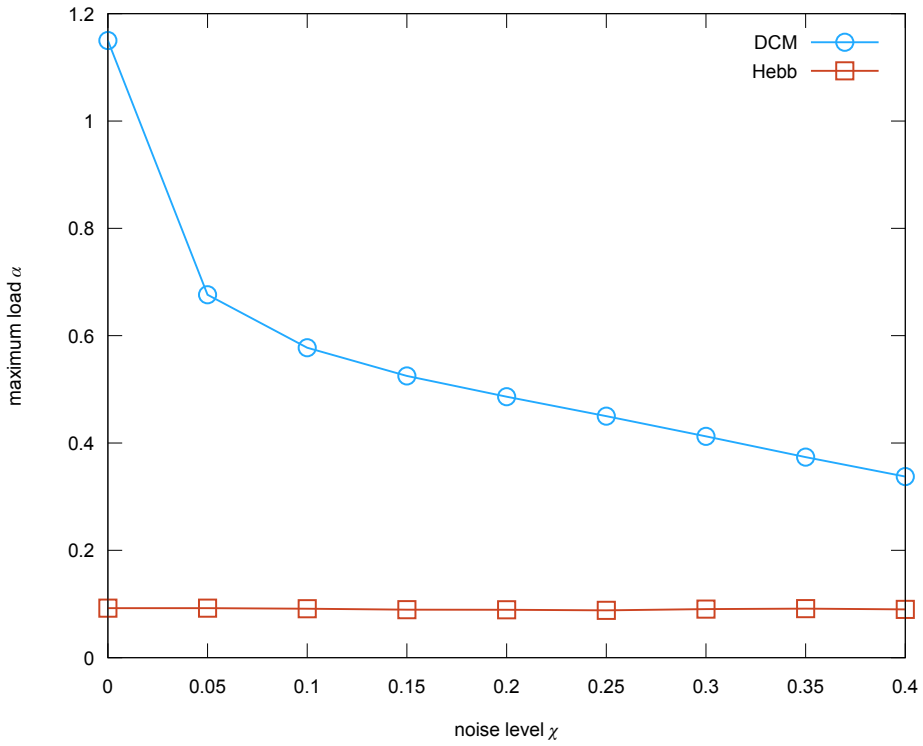

Fig. 4.2 Storage load as a function of the noise level. The behaviour of the DCM learning rule (blue curve) and the one of Hebbian learning (red curve) are averaged over 10 samples. The error bars are not visible at the plot resolution. In the simulation $N = 401$ and $\beta = 2$.

4.2.1 Weak external fields

The case of strong external signals is not consistent from a biological point of view. Indeed, high intensities of the external field bias the neuronal activity towards the inputs, so that the dynamics of the network is fully dominated by the incoming stimulus. On the other hand, weak intensities make the evolution in time of the neuronal activity to be entirely ruled by the noisy recurrent stimulus, exerted on a given neuron by means of the surrounding nerve cells. One important issue is then

how the intensity of the external field has to be chosen in order to balance the two different effects.

To this end, we need to consider that the recurrent stimulus relies on the strength of the synaptic connections. However, as long as the network learns new patterns, the synaptic connections get stronger and stronger. The intensity of the external field has then to grow as well, in order to compensate the increasingly strength of the recurrent stimulus. This effect is shown in Fig. (4.3), where we plot the maximum achievable storage load as a function of the external field intensity. Notice that, the critical capacity quite quickly saturates to the asymptotic value (dashed line), estimated for infinite external fields. In particular, as the inset in Fig. (4.3) shows, these values of the maximum storage load are observable for intensities of the external fields slightly smaller than the ones of the recurrent stimulus. Therefore, weak external signals seem to be already sufficient for achieving good learning performances, as already suggested [114].

4.2.2 Sparse neural coding and Dale's law

A series of experimental and theoretical evidences have pointed out that, in biological networks, the number of simultaneously active neurons is just a fraction of the entire population. This suggests that the neural coding is actually sparse. Sparse coding can improve the processing of information within the brain. For instance, it saves metabolic energy and increases the memory capacity, reducing the cross-talk between the different patterns of neuronal activity ([83] and references therein).

To this end, in the following, we will consider the neuronal states to be described by $s_i \in \{0,1\}$ binary variables, thus avoiding the implausible feature of symmetry between the active and the quiescent state of the $s_i \in \{-1,1\}$ binary model. The sparsity or coding level f is then defined as the average fraction of simultaneously active neurons, namely $f = \frac{1}{N}\sum_{i=1}^{N} s_i$.

In order to make this scenario even more consistent from a biological point of view, we introduce a further biological constraint, imposed by the Dale's principle. According to the Dale's principle, a neuron can be either excitatory (positive synaptic couplings) or inhibitory (negative synaptic couplings) [80, 81]. This is known to reduce the critical capacity mostly up to the half [115].

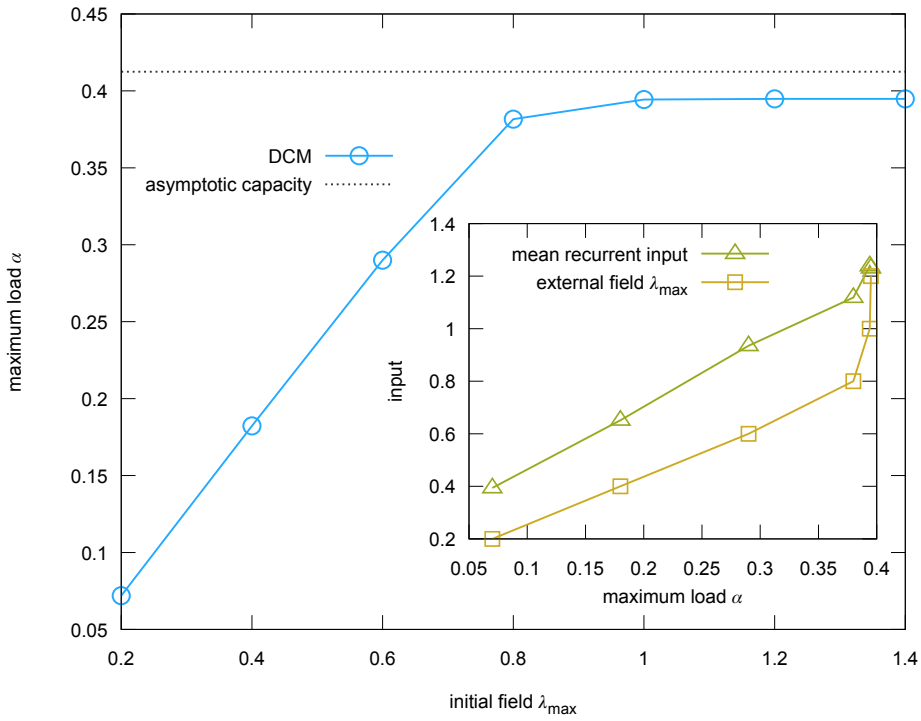

Fig. 4.3 Critical capacity as a function of the external field intensity. In the simulation, $N = 401$, $T = 20$ and $\Delta\lambda = \lambda_{max}/3$, $\beta = 2$. In the retrieval phase the noise level is set at $\chi = 0.3$. In the inset, external field intensity (yellow curve) and recurrent stimulus from surrounding neurons (green curve) as a function of the maximum storage load. The curves are averaged over 100 samples.

For the sake of simplicity, in this work, only the synaptic couplings of the excitatory neurons undergo through synaptic plasticity. The inhibitory network is instead replaced by three different inhibitory schemes, designed for miming more complex inhibitory phenomena. Their task is to regulate the current activity of the network, namely $S^t = \frac{1}{N}\sum_{i=1}^{N} s_i^t$, according to the desired coding level f. This prevents the excitatory network to converge towards the trivial states of all on or all off neurons. The details concerning the analytic derivation of the three schemes are provided in Appendix C.

In the first inhibitory scheme, the role of the inhibitory network is played by a global inhibitory unit, which set the activity of the network to the desired coding level, relying on an elastic feed-back signal [116]. The second inhibitory scheme has to be thought as a soft "winner-takes-all" mechanism [117–127]. In this case, an inhibitory signal is triggered by the active neurons when their number reaches the desired activity level. The signal then sets down the local fields of the neurons in the quiescent state, thus avoiding their activation. The last inhibitory scheme is based on the widely observed phenomenon of threshold variability in the central nervous system [128]. It has been shown that, in presence of highly correlated stimuli, threshold variability is fundamental for enhancing neural computation [115]. Actually, as we will see, this is also the case in our learning model, where threshold variability turns out to be a necessary requirement for allowing the storage of an extensive number of patterns in the on-line learning regime.

Fig. (4.4) shows the maximum achievable storage load α as a function of the number of epochs required for storing a number $p = \alpha N$ patterns, for both synaptic couplings constrained to satisfy the Dale's principle (red curves) and unconstrained synaptic couplings (blue curves). We can clearly notice that, the introduction of the biological constraints imposed by the sparseness of the neural coding and the Dale's principle, only slightly decreases the maximum achievable storage load, detectable when the number of required learning cycles starts diverging.

4.2.3 The DCM in comparison with the Hebbian learning

In this section, we want to compare the DCM learning rule with Hebbian learning for what concerns biased and correlated patterns, spurious attractors and on-line learning settings.

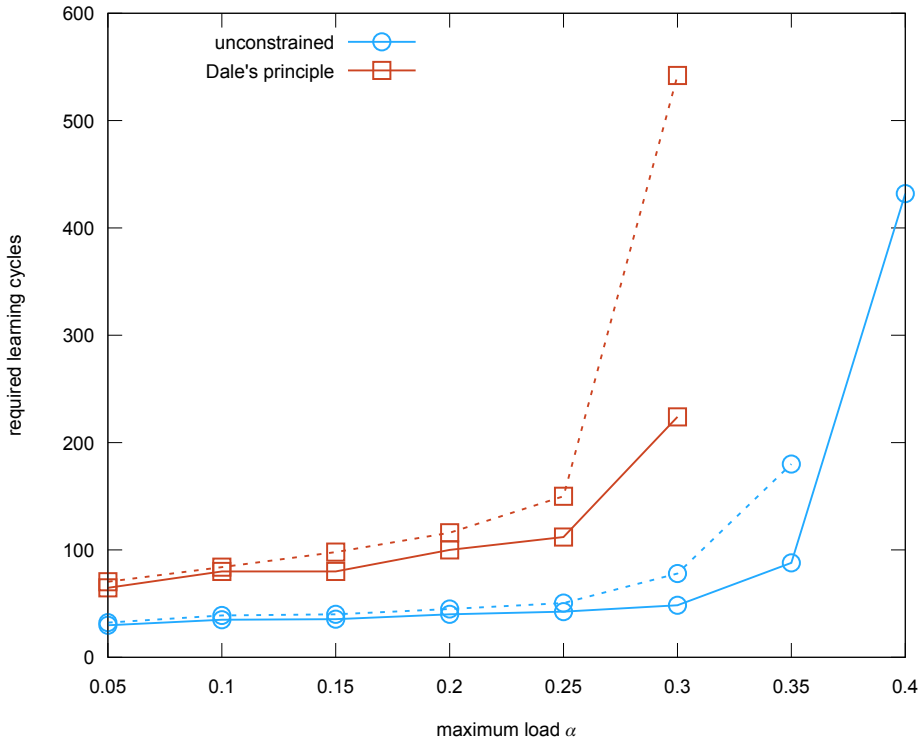

Fig. 4.4 Required number of epochs versus the maximum storage load. The dashed curve refers to $N = 200$, while the thick curve is obtained for $N = 400$. In the simulation, the level of noise in the retrieval phase is $\chi = 0.3$ while the coding level is $f = 0.5$. The exploited inhibitory scheme is the one of the soft winner-takes-all mechanism.

Correlated and Biased patterns

The sensory inputs from the external world are typically not only sparse but also redundant. A learning model has then to be able to deal with correlated input signals, in order to describe more realistic scenarios.

As anticipated in the first chapter, one way of introducing correlations among patterns of neuronal activity is to sample them according to a biased probability distribution (see Eq. (1.53)) [69]. Fig. 4.5 shows the maximum achievable capacity as a function of the pattern bias. We can clearly notice that the DCM learning rule (blue curves) achieves higher critical capacities with respect to Hebbian learning (red and green curves). Moreover, as pointed out in the first chapter, when dealing with biased patterns, Hebbian learning has to be modified (green curve) in light of the pattern bias (see Eq. (1.54)) [69]. Notice that, the not vanishing critical capacity in the case of standard Hebbian learning (red curves), is due to finite size effects as well

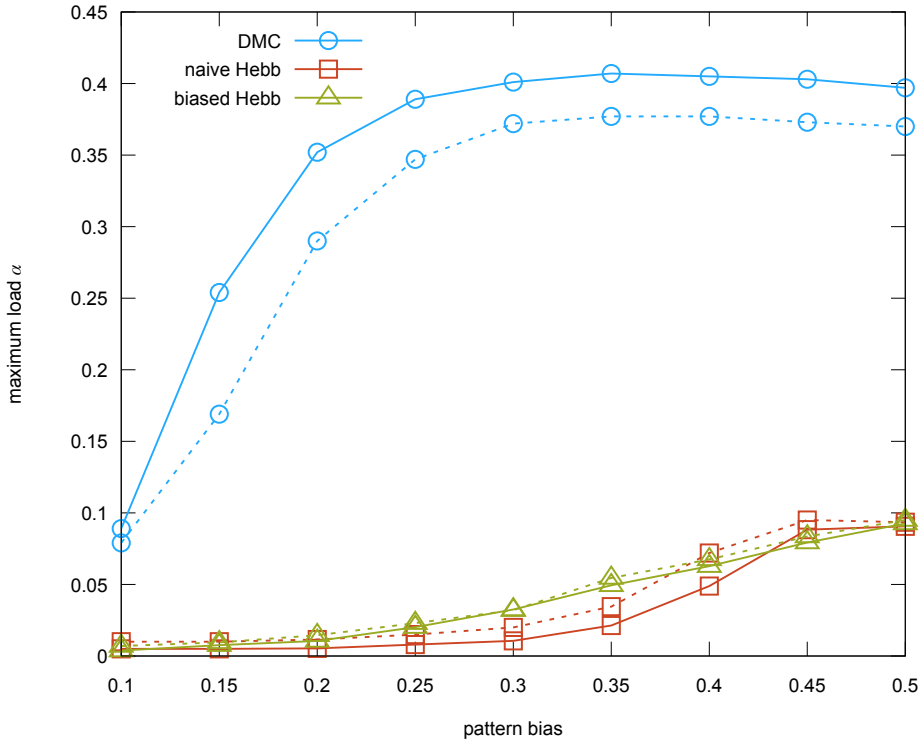

Fig. 4.5 Maximum storage load as a function of the pattern bias. The dashed and the thick curves refer to different network sizes, respectively $N = 200$ and $N = 400$. The blue curves are relative to the DCM learning rule, whereas the red and the green curves refers respectively to standard Hebbian learning and biased Hebbian learning. In the simulations, the results are averaged over 10 samples. The error bars are not detectable at the resolution level of the plot.

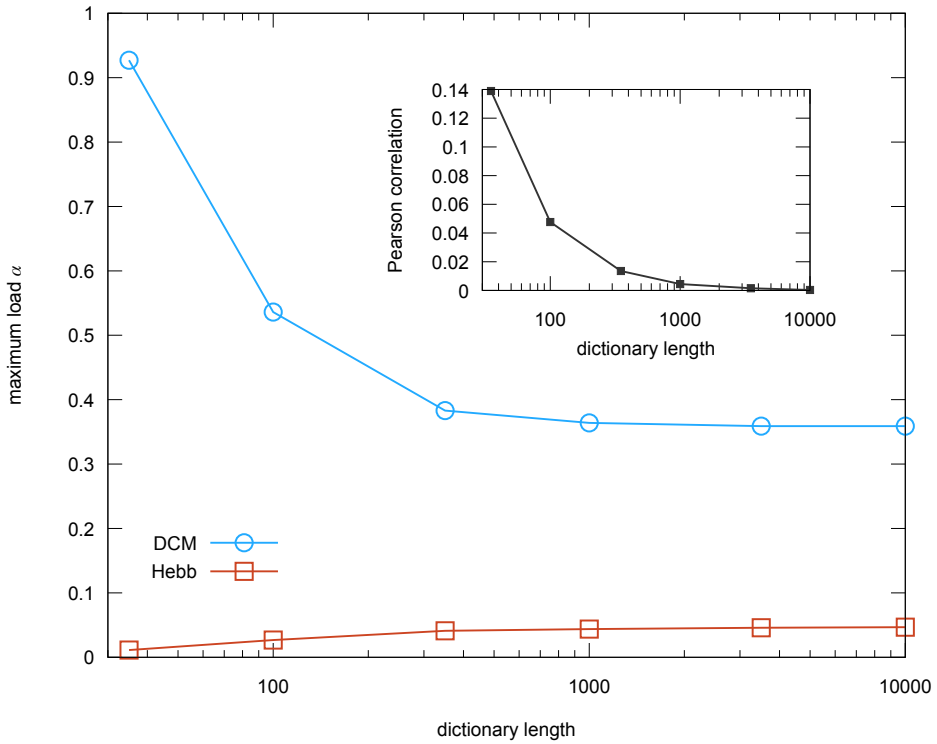

Fig. 4.6 Maximum storage load as a function of the dictionary length. The blue curve is relative to the DCM learning rule, whereas the red one refers to Hebbian learning. In the simulation, the sparsity level of each feature, namely f, is set at 0.1. The patterns are obtained as a linear combination of 6 features out of the 200 contained in the dictionary. The inset shows the mean Pearson correlation as a function of the dictionary length. In the simulation, the results are averaged over 10 samples. The error bars are not detectable at the resolution level of the plot.

as the vanishing maximum storage load for small biases of the modified Hebbian learning.

Another way of introducing correlations among patterns is to write them as a linear combination of sparse vectors Υ, called features and sampled according to the distribution $P(\Upsilon_i) = f\,\delta(\Upsilon_i - 1) + (1-f)\,\delta(\Upsilon_i)$. To this end, we first generate a dictionary of L features, namely $D = \{\Upsilon^\nu\}_{\nu=1}^{L}$. We then construct each pattern as a linear combination of a certain number of features, namely $\xi_i^\mu = \Theta\left(\sum_{\nu=1}^{L} c_\nu^\mu \Upsilon_i^\nu\right)$, where $c_\nu^\mu \in \{0,1\}$ are the coefficients of the linear combination, selecting a specific feature in the dictionary, and $\Theta(x)$ is the Heaviside step function, acting as the logic OR operator.

Fig. (4.6) shows the maximum achievable storage load as a function of the dictionary length. We can clearly see that the DCM learning rule (blue curve) reaches higher critical storage loads compared to Hebbian learning (red curve), further profiting of the higher correlation level for smaller dictionary lengths. The inset provides a measure of the correlations among patterns as a function of the dictionary length, measured in terms of the mean Pearson correlation.

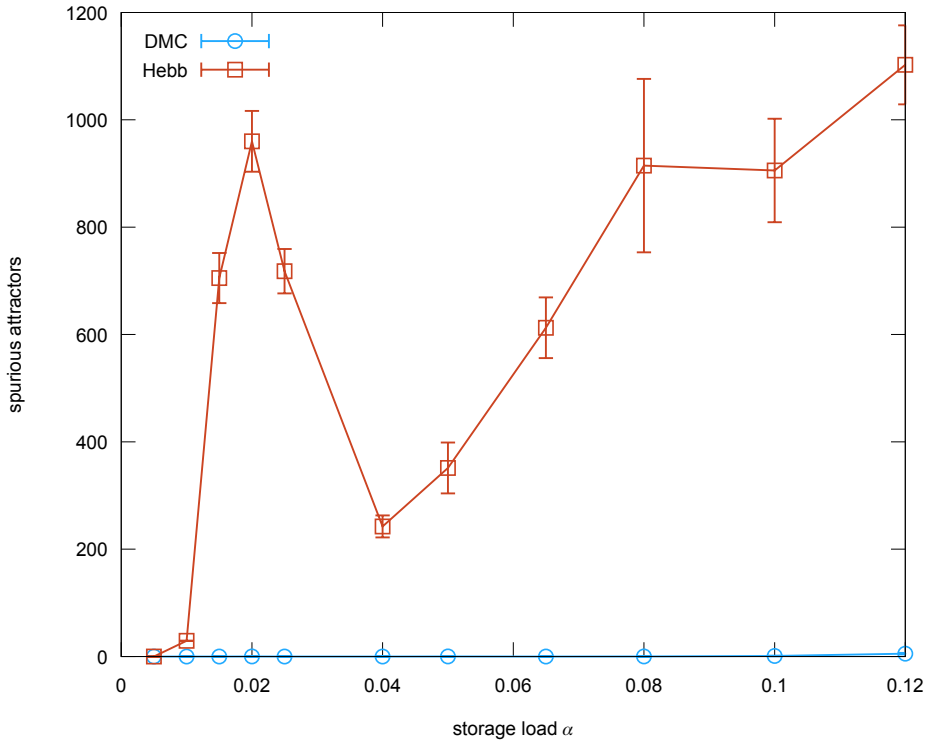

Fig. 4.7 Number of spurious attractors as a function of the storage load. In the plot, we compare the number of detected spurious attractors for both the DCM learning rule (blue curve) and Hebbian learning (red curve). We identify them through 1000 restarts of the network evolution, made of 200 steps of Glauber dynamics. In the simulation, the size of the network is $N = 400$, and the results are averaged over 10 samples.

Spurious Attractors

One of the limits of Hebbian learning, affecting the retrieval of stored patterns, is the presence of spurious attractor states. Indeed, because the Hopfield Hamiltonian is invariant with respect to the change of sign of the neuronal state variables, not only the patterns we want to store are minima of the Hopfield Hamiltonian but also the

ones obtained reversing the patterns sign. More generally, any linear combination of the patterns is itself a stable minimum of the Hopfield Hamiltonian. The spurious states are also called mixture states, since they typically lie at the intersection between two distinct basins of attraction, relative to two different patterns [64].

Fig. 4.7 shows the number of spurious attractors as a function of the storage load (see Appendix C for the details concerning the numerical simulation). We can clearly notice that, contrary to Hebbian learning (red curve), the DCM learning rule seems to be not affected by spurious attractors, thus being capable of preventing the basins of attraction from overlapping. The first peak of Hebbian learning can be explained as a finite size expression of the actual exponential growth of spurious states in the sub-extensive regime, namely very small storage loads [129]. In the extensive regime, namely higher storage loads, linear combinations of a large number of patterns start disappearing, so that they are no more exponentially many. In this case, only mixtures of odd numbers of patterns survive [129].

4.2.4 One-Shot learning

In the on-line inspired learning scenario described in this work, the input patterns are presented to the network one at a time. We switch to the next pattern only when the previous one has been already properly stored. The patterns that have been correctly retrieved are never shown again to the network. After a certain number of iterations, we expect this learning regime to reach a steady state, where the number of stored patterns does not grow any more. This basically because the previously stored patterns give way to the newly stored ones. The capacity at which this stationary condition is reached is known as *palimpsest capacity* [130].

In this setting, considering $s_i \in \{-1, 1\}$ binary neuronal state models, the DCM learning rule is able to store an extensive number of input patterns, reaching the palimpsest capacity $\alpha_p \simeq 0.05$. This result is comparable with other learning rules proposed for on-line learning regimes [130–132]. The same holds if we consider the sparse neuronal model $s_i \in \{0, 1\}$. However, in this case, for this to be true we need to further take into account threshold variability (see Appendix C).

Fig. 4.8 shows the number of stored patterns as a function of the total number of patterns that we wish the network to store, for different network sizes. We can clearly notice that after a transient regime, where all the input patterns are correctly

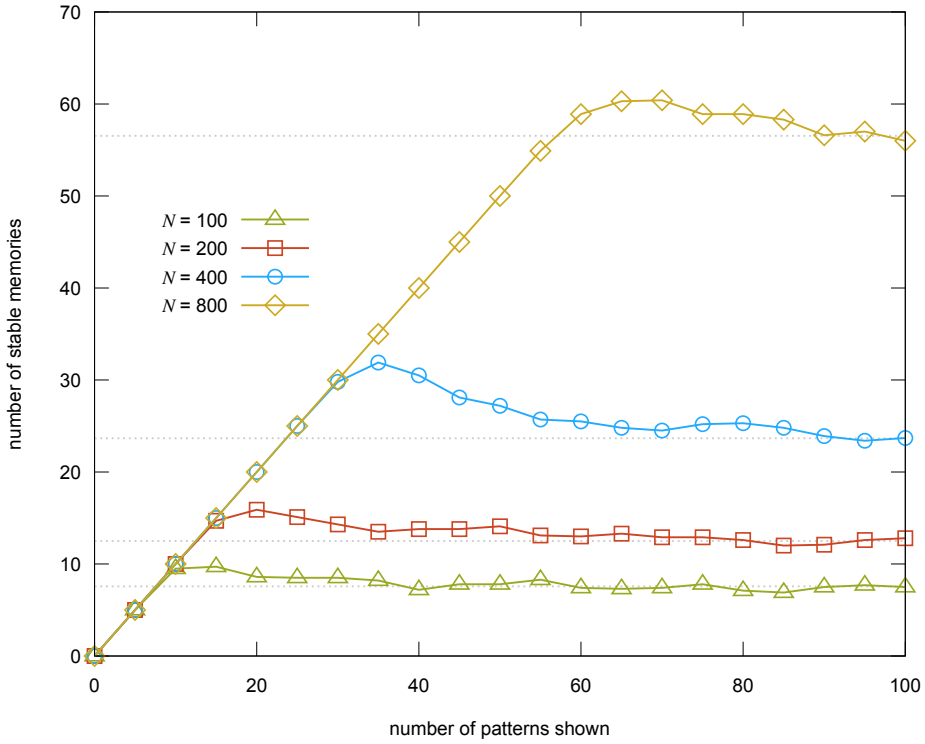

Fig. 4.8 Scaling of the palimpsest capacity with the network size. In the plot we show the number of stored patterns as a function of the total number we want to store within the network. The different curves refer to different network sizes, namely $N = 100,\ 200,\ 400,\ 800$. The neuronal state model here considered is $s_i \in \{-1, 1\}$. The dashed gray curve are obtained as an average over the last 3 measurements and the results are averaged over 10 samples.

retrieved, the learning rule reaches a stationary condition when approaching the palimpsest capacity.

4.3 Visible-Hidden

In the previous section, we have considered the specific case of a recurrent neural network, made of solely visible units. In this section, we instead focus on the more general case of a recurrent neural network when endowed of additional hidden neuronal states. As pointed out in the first chapter, the introduction of hidden units has the effect of increasing the representational power of the network, by allowing it to catch more complex correlations in the data structure. In this case, the network is then able to infer the statistics of the external stimulus and then to generate new data samples according to it, thus acting as a generative model.

The learning model proposed in this chapter, can be equivalently applied to this context. In particular, in the following, we will consider the specific case of a recurrent neural network where the synaptic connections are restricted to the solely visible-hidden units. As said, this model goes under the name of *Restricted Boltzmann Machine* (RBM).

The DCM learning rule does not require any further adjustment or restatement in this case. It is always derived through the optimization of the KL divergence in Eq. (4.2). In particular, in the specific case of strong biasing signals, we can explicitly write the expression of the probability distribution over all the possible initial neuronal states in Eq. (4.3). Indeed, since in RBMs the connections are restricted to the solely visible-hidden units, the distribution over the hidden states factorizes at fixed configurations of the visible ones:

$$P_{\text{clamp}}\left(\boldsymbol{s}_V, \boldsymbol{s}_H; \boldsymbol{\xi}\right) = \prod_{i \in V} \delta_{s_i, \xi_i} \prod_{j \in H} P\left(s_j | \boldsymbol{s}_V = \boldsymbol{\xi}\right). \tag{4.10}$$

The synaptic couplings are then adapted according to the resulting DCM learning rule:

$$\Delta W_{ij} \propto P\left(s_j|\boldsymbol{s}_V = \boldsymbol{\xi}\right)\xi_i s_j - \sum_{\{s_{k\in H}\}}\prod_{k\in H} P\left(s_k|\boldsymbol{s}_V = \boldsymbol{\xi}\right) P\left(s'_i|\boldsymbol{s}_H\right) s'_i s_j, \tag{4.11}$$

with $i \in V$ and $j \in H$. As pointed out in the case of fully-visible networks, even in this case, the optimization of the KL divergence can be translated into an on-line optimization of the log-pseudo-likelihood:

$$\begin{aligned}&\mathscr{L}\left(\{\boldsymbol{\xi}^\mu\}|W_{ij},\theta;\beta\right) = \\ &= \frac{1}{M}\sum_{\mu=1}^{M}\sum_{i\in V}\log\left(\sum_{s_{k\in H}}\prod_{k\in H} P\left(s_k|\boldsymbol{s}_V = \boldsymbol{\xi}\right) P\left(s'_i = \xi_i^\mu|\boldsymbol{s}_H;\lambda^{ext}=0\right)\right).\end{aligned} \tag{4.12}$$

The two optimization problems are then equivalent and lead two the same learning rule in Eq. (4.11). The only difference with respect to the fully-visible case, is that the synaptic couplings have now to be inferred from a set of incomplete data. Indeed, only the visible units are subject to the external stimulus.

The updated rule of Eq. (4.11) is quite close to Contrastive Divergence in Eq. (1.71). Indeed, even in this case we can distinguish between a positive phase, where the state of the visible units is clamped on the input data, and a negative phase, where instead the configurations of the neuronal states need to be sampled according to Gibbs chains. We thus believe that our derivation can shed more lights on the effectiveness of Contrastive Divergence, even when single samples of the Gibbs chain are performed, by directly initializing the chain on the input data.

However, we need here to point out that, while Contrastive Divergence has been designed for energy-based generative models, where the symmetry of synaptic couplings is a crucial requirement for the definition of an Hamiltonian, the DCM rule is completely asymmetric and, therefore, legitimately applicable to the context of kinetic models.

Fig. 4.9a shows the features extracted by the hidden units from the images of hand-written digits in the MNIST data set [11], according to the learning rule in Eq. (4.11), namely in the case of strong biasing signals suddenly dropped to

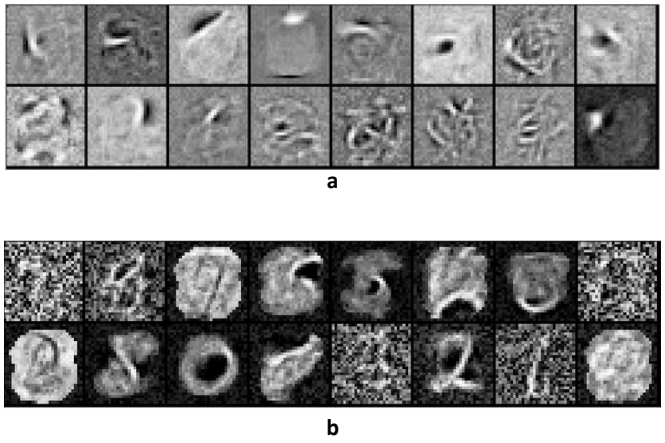

Fig. 4.9 Features extraction. The plot makes a comparison between the features extracted in the case of strong biasing signal suddenly depressed, where the correlations are estimated by means of the TAP approach (a), and the features extracted when considering the more biological plausible scenario of weak external fields progressively vanishing, where the correlations are directly measured from the network dynamics. In this case, we exploit the $s_i \in \{0,1\}$ neuronal state model, together with the soft winner-takes-all inhibitory scheme.

zero. The ensemble average in Eq. (4.11) is estimated by means of the mean field Thouless-Anderson-Palmer (TAP) approach [133], that has been shown to hold in the context of kinetic models [110–112]. In Appendix C, we provide the extension of the derivation of the TAP equations for kinetic models of Ref. [110], to the case of $s_i \in \{0,1\}$ binary neuronal state models. This learning scenario allows for a direct comparison with the performance achieved by the Contrastive Divergence algorithm in RBMs training. Fig. (4.9)b shows instead the features extracted by the hidden units in a more biologically plausible learning scenario. In this case, the input patterns are presented to the network as external fields of weak intensity and the correlations are directly estimated from the dynamics of the network, simulated through the Glauber stochastic process. Moreover, we introduce the further constraints of sparse neural coding and the one imposed by the Dale's law. In this case, we choose the soft-winner-takes all as the inhibitory mechanism, aiming to mimic the underlying effect of the inhibitory network.

The ability of the network to generate new data sample according to the statistics of the input data, is shown instead in Fig. 4.10. The two sets of samples refer respectively to the case of strong biasing signals (a) and the more biologically inspired learning scenario (b). We can clearly notice that, the introduction of the biological

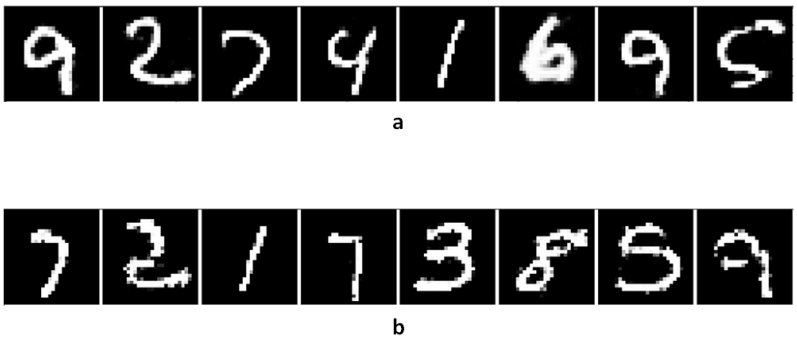

Fig. 4.10 Samples generations. The plot shows the sample generated after 100 time-steps of Glauber dynamics, when initialized on 8 different states of neuronal activity. The samples are readable as the probability associated to the final configuration of the visible units, at the end of the dynamics. (a) Case of strong biasing external signals suddenly vanishing. The correlations are estimated through the TAP approach with no inhibitory schemes. In the simulation, $\beta = 2$. (b) Case of weak external signals progressively vanishing. The correlations are directly estimated from the dynamics and the inhibitory scheme employed is the one of soft winner-takes-all. In the simulation, $\beta = 30$. The higher resolution in (a) has to be addressed to the different choice of the inverse temperature parameter in the two simulations.

constraints seems to not consistently affect the performances of the network as a generative model.

Finally, we have compared the performances achieved by the network in classification tasks, when considering both the two above mentioned learning scenarios. In a classification task, the RBM is required to recognize which number is depicted in the image, thus assigning to each image its corresponding digit. To this end, we add a further layer of 10 neurons, one for each digit, where to read the response of the network. During the training, one of the ten neurons receives an external supervisory signal, in the form of an external field, whenever the corresponding digit is shown to the network as input signal [134].

The generalization error turns out to be $\varepsilon_g \simeq 2.76\%$ in the case of strong biasing signals, and $\varepsilon_g \simeq 7.74\%$ in the case of the more biological inspired learning scenario. These values need to be compared with the ones achieved by Contrastive Divergence, namely $\varepsilon_g \simeq 0.3\%$ [11]. The discrepancy between the two results has to be addressed to the smaller size of the network involved in the classification task, and to the addition of stochastic components. These results should then not be disregarded,

especially if we consider that they have been obtained in a more biologically setting, where the introduction of some biological constraints inevitably affects the cognitive abilities of the network. Moreover, we should also take into account that we have not employed a proper supervisory signal. More details on the numerical simulations in this section are provided in Appendix C.

Chapter 5

Conclusion, Discussions and Further Perspectives

It is strongly believed that phenomena like learning and memory emerge when neurons are connected with each other, constituting a complex network of interactions. In the same vein of what happens in other physical systems, characterized by many interacting particles, these collective phenomena are typically studied by means of methods borrowed from statistical physics and relying on simplified models of single neurons [6].

This work investigates neural networks from the perspective of statistical physics. It is mainly focused on the two extreme topologies of neural networks, namely feed-forward and recurrent neural networks.

In feed-forward neural networks, the input signal propagates forward from the input layer up to the output one. The topology of this kind of networks make them particularly suitable in implementing classification tasks. They are thus typically employed as *discriminative models*, whose aim is to gather a given number of objects in different categories, on the basis of their characteristic features. This condition can be eventually achieved at the end of a training phase, during which the synaptic couplings of the network are adapted in order to assign the object to its corresponding class. Learning can then be thought as a search in the space of synaptic states, looking for those configurations of the synaptic couplings satisfying the classification task and actually representing the *solutions* of the learning problem [8].

In this regard, we have mainly focused on the simplest kind of feed-forward neural networks, namely the *Perceptron*. After having introduced the spherical Perceptron model, namely when synaptic couplings are assumed to be continuous variables confined on an hyper-sphere, and the binary Perceptron model, namely when synaptic couplings are instead thought as binary variables, we have investigated the more general and realistic case of the Discrete Perceptron model, where a finite and discrete number of synaptic states are actually employed for encoding information and where the sparseness of the neural coding, as well as, the Dale's principle are taken into account.

To this end, we extended both the analysis of the typical learning scenario [56] and the corresponding large deviation analysis [78] of binary Perceptron models, to this more general setting. It turned out that the same qualitative scenario holds for the two models: typical solutions are isolated in the version space and embedded in a landscape rich of meta-stable synaptic states. Despite that, there exist regions of the version space, not detectable in the typical learning scenario described by the Gibbs measure, but revealable thanks to the definition of an out-of-equilibrium measure, that re-weights the statistical weight of the synaptic configurations in light of the number of solutions surrounding a reference one. Within these regions, solutions are organized in extremely dense clusters.

Thanks to the replica formalism, we have been able to detect for which values of the storage load, the clusters are still guarantee to exist. In particular, we managed to determine the specific value of the storage load at which the cluster is supposed to either disappear, or to break in several disconnected components. This value has always been found above $0.9\alpha_c$, with α_c being the critical capacity of the model, namely the one above which no more solutions can be detected. This picture turned out to be really robust to the tuning of both the number of available synaptic states and the sparsity level.

We need here to point out one important further result. The critical capacity of the discrete Perceptron model quite quickly saturates to the asymptotic limit of continuous synaptic couplings. This suggests that synapses may exploit very few bits of precision for encoding information [79].

One of the aim of the statistical physics approach to discrete Perceptron models, was also to prove that the existence of extremely dense and robust clusters of solutions, is still preserved in more biologically plausible settings. From the biological

point of view, the discrete Perceptron model proposed in this work, is more realistic with respect to the binary one, although still very far from describing real learning scenarios, where typically input-output correlations have to be taken into account. In spite of this, we have anyway show that the existence of extreme dense clusters of solutions seems to be a characteristic trait of discrete networks, not affected by the specific details of the model.

Moreover, despite the fact that the statistical physics analysis in the second chapter, has been performed on a simple model, it can anyway provide more insights on more biologically plausible ones. Indeed, it corroborates the hypothesis, suggested by several experiments [40–42], of synaptic connections as binary switch devices, providing at least one reason why this should be the actual strategy implemented by the brain: the existence of extremely dense clusters provides regions of the version space, where learning can take place in a much more efficient way, especially in terms of robustness to noise.

This has indeed inspired the design of new learning algorithms as solvers of the discrete Perceptron models, like the Entropy Driven Monte Carlo, and clarified the reasons determining the effectiveness of the already existing learning protocols, based on message-passing.

A still open question is the one concerning the role played by stochasticity in neural computation. In the third chapter, we have thus proposed a binary Perceptron model, where synapses are considered as intrinsically stochastic components. In this setting, we have shown that stochasticity naturally drives the learning process towards extremely dense regions of solutions. We have proved this to be the case both from a theoretical point of view, by setting up a statistical physics analysis based on the replica formalism, and from an algorithmic point of view, by designing a new learning algorithm, able to translate the binary learning problem into a continuous optimization problem. We thus hope that the proposed learning model can provide more insights on learning in biological networks, where stochasticity and low precision in synaptic connections are currently matter of a lively debate.

Through an analysis concerning the geometric properties of the version space in stochastic binary Perceptrons models, we have further highlighted the connection between the flatness of the energy landscape in dense regions of solutions and the generalization performances, by comparing the new learning algorithm, and its more

biologically inspired variants, with both Simulated Annealing [135] and Reinforced Belief Propagation [57].

As a next step, we have recently started to extend the stochastic learning model of the third chapter, to deep learning settings. We have trained a feed-forward neural network with three layers of hidden units by means of the backpropagation algorithm, employing the stochastic gradient descent schedule and exploiting an approximation of uncorrelated neurons [136]. We have already obtained $\sim 1.7\%$ as generalization error on the MNIST data set [99]. This is an encouraging result, if we consider that in deep learning synaptic couplings are usually continuous and convolutional layers are typically employed for image recognition. Moreover, we should also notice that, the previously proposed learning algorithms for binary Perceptrons have never been extended to deep learning, and that the newly proposed learning model can be straightforwardly generalized to other constraint satisfaction problems.

In the fourth chapter, we have mainly focused on recurrent neural networks, namely those networks where the response of each neuron plays the role of the incoming signal for the other neurons to whom it is connected. As seen, these networks are typically exploited as devices for the storage of specific patterns of neuronal activity as well as *generative models*, namely those models able to infer the statistic of the input data and then to generate new data sample according to the inferred statistics.

In this context, we have proposed a new learning rule, according to which the synaptic couplings have to adapt, in order to make the network able to store a given set of patterns, or, more generally, to behave as a generative model. This new learning rule, which we have called *Delayed Correlation Matching*, arises from the optimization of the KL divergence between the dynamics of the network in presence of an external field of intensity λ_1 and carrying information on the input data, and the dynamics of the same network but in presence of a weaker external field λ_2. In the case of strong biasing signals suddenly vanishing, we have seen that optimizing the KL divergence is actually equivalent to optimize an on-line log-pseudo-likelihood.

Moreover, we have further shown that all the information needed to adapt the synaptic couplings can be directly deduced from the dynamics itself. Indeed, the DCM learning rule relies on a matching condition between time-delayed correlations in presence of different field intensities, which aims to absorb the driving effect of the external field within the synaptic connections.

The DCM learning rule shows some common features as well as some distinct traits with both Hebbian learning [66] and Contrastive Divergence [77]. Indeed, all the three learning rules update the synaptic connections exploiting only information that are locally available to each synapse. In particular, Hebbian learning can be derived as the limit of strong biasing signals of the DCM learning rule. However, we need here to point out that, conversely to the other two learning rules, DCM preserves the asymmetry of synaptic couplings, thus being applicable to the context of kinetic models.

The ultimate goal of this work was to design a new learning rule while relying on a series of biological constraints, which can make the learning rule more amenable from a biological point of view. We have thus considered the case of weak external fields, that prevent the dynamics of the network from being fully dominated by the external stimulus. We have further taken into account the sparsity of the neural coding and the further constraint imposed by the Dale's law, according to which we can identify two different neuronal populations, namely the excitatory and the inhibitory neurons [80, 81]. In particular, we have proposed three different inhibitory schemes, aiming to mimic the effect of the underlying inhibitory network. We have shown that, the introduction of all these biological constraints does not consistently affect the network performances in terms of maximum achievable storage loads. This still holds when adding hidden neuronal states: the DCM learning rule preserves the representational power of the network, even in presence of the above-mentioned biological constraints.

The DCM learning rule is further able to store patterns of activity in an online learning regime, surprisingly reaching an extensive palimpsest capacity, and to deal with correlated input patterns.

However, we should here to point out that matching two different quantities in presence and in absence of an external stimulus, and then exploiting this matching for learning, is not a new idea. This intuition has been already exploited in the context of both discrete-time dynamics [137] and spiking neural networks [138, 139], where the matching quantities are typically local currents, rather than correlations among neuronal states [140–142]. However, despite the fact that these models are more detailed and closer to biology, they typically rely on not local learning protocols.

The same idea of matching correlations has been already developed in Ref. [114]. However, in this case, the learning rule can only provide symmetric synaptic

couplings, being based on the matching between equal times correlations, estimated through Belief Propagation.

The model in Ref. [114] is relative to diluted recurrent neural networks, namely those networks where each neuron is connected to just a subset of all the other neurons in the network. Recently, we have adapted the same model to the case of fully connected neural networks [143]. In particular we have shown that, in presence of highly correlated patterns of activity, the performances of the resulting learning rule can be enhanced if, instead of matching the correlations in presence of an initial external field of intensity λ_1 with the ones in presence of a vanishing field λ_2, we match the first ones with the correlations estimated in presence of an antagonist external field of intensity $-\lambda_1$. Why this is the case can be explained considering that, the last matching not only requires the synaptic connections to absorb the driving effect of the external field, but also to react to an external stimulus trying to detrimentally interfere with the storage of the input patterns. In this way we are thus introducing some robustness within the model. As a next step, we will then apply this idea to the DCM learning rule, in order to see if this can further improve the learning performances in presence of correlated patterns.

In Ref. [116], the authors designed a three threshold learning rule based on biological features and relying on the distribution of the input data, instead of the evolution of the network. However, conversely to what happens when applying the DCM learning rule in the context of weak external fields, it would seems that the three threshold learning rule needs stronger external signals, being the storage load drastically decreasing with the external fields.

In the next future, it would be interesting to generalize this learning model to continuous time scales, spiking networks and time-dependent inputs. Moreover, more complex architectures of visible and hidden neurons than RBMs should be taken into account.

References

[1] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[2] S. Seung. *Connectome: how the brain's wiring makes us who we are*. Houghton Mifflin Harcourt, 2012.

[3] L. Lapicque. Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization. *Journal of Physiol Pathol Générale*, 9:620–635, 1907.

[4] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol.*, 117(4):500–544, 1952.

[5] P. Dayan and L.F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural System*. The MIT press, Cambridge, Massachusetts, London, England, 2001.

[6] A. Engel and C. Van den Broeck. *Statistical Mechanics of Learning*. Cambridge University Press, 2004.

[7] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *bulletin of mathematical biophysics*, 5, 1943.

[8] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[9] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[10] F. Chollet. *Deep learning with Python*. Manning Publications, 2018.

[11] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[12] A. Krizhevsky, V. Nair, and G. Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 2014.

[13] T.V.P. Bliss, G. L. Collingridge, et al. A synaptic model of memory: long-term potentiation in the hippocampus. *Nature*, 361(6407):31–39, 1993.

[14] M. T. Rogan, U. V. Stäubli, and J. E. LeDoux. Fear conditioning induces associative long-term potentiation in the amygdala. *Nature*, 390(6660):604–607, 1997.

[15] J. R. Whitlock, A. J. Heynen, M. G. Shuler, and M. F. Bear. Learning induces long-term potentiation in the hippocampus. *science*, 313(5790):1093–1097, 2006.

[16] L. Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[18] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[19] P. Chaudhari, A. Choromanska, S. Soatto, and Y. LeCun. Entropy-sgd: Biasing gradient descent into wide valleys. *arXiv preprint arXiv:1611.01838*, 2016.

[20] P Carnevali. P. carnevali and s. patarnello, europhys. lett. 4, 1199 (1987). *Europhys. Lett.*, 4:1199, 1987.

[21] P. Del Giudice, S. Franz, and M.A. Virasoro. Perceptron beyond the limit of capacity. *Journal de Physique*, 50(2):121–134, 1989.

[22] H.S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Physical Review A*, 45(8):6056, 1992.

[23] K. Binder and A. P. Young. Spin glasses: Experimental facts, theoretical concepts, and open questions. *Reviews of Modern physics*, 58(4):801, 1986.

[24] M. Mézard, G. Parisi, and M. Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

[25] F. Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, CORNELL AERONAUTICAL LAB INC BUFFALO NY, 1961.

[26] M. Opper and W. Kinzel. Statistical mechanics of generalization. In *Models of neural networks III*, pages 151–209. Springer, 1996.

[27] M. Minsky and S. Papert. Perceptron (expanded edition), 1969.

[28] M. Bouten, J. Schietse, and C. Van den Broeck. Gradient descent learning in perceptrons: A review of its possibilities. *Physical Review E*, 52(2):1958, 1995.

[29] T. Castellani and A. Cavagna. Spin-glass theory for pedestrians. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(05):P05012, 2005.

[30] E. Gardner. The space of interactions in neural network models. *Journal of physics A: Mathematical and general*, 21(1):257, 1988.

[31] E. Gardner and B. Derrida. Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and general*, 21(1):271, 1988.

[32] E. Gardner. Optimal basins of attraction in randomly sparse neural network models. *Journal of Physics A: Mathematical and General*, 22(12):1969, 1989.

[33] E. Gardner and B. Derrida. Three unfinished works on the optimal storage capacity of networks. *Journal of Physics A: Mathematical and General*, 22(12):1983, 1989.

[34] G. Gyorgyi and N. Tishby. In *Neural Networks and Spin Glasses*, pages 3–36. edited by W.K. Theumann and R. Köeberle, (World Scientific, Singapore), 1990.

[35] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the theory of neural computation.* Addison-Wesley/Addison Wesley Longman, 1991.

[36] T.L.H. Watkin, A. Rau, and M. Biehl. The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65(2):499, 1993.

[37] C. Broeck. Statistical physics of learning from examples: a brief introduction. *Acta Physica Polonica. Series B*, 25(6):903–923, 1994.

[38] B. Widrow and M. E. Hoff. Adaptive switching circuits. Technical report, STANFORD UNIV CA STANFORD ELECTRONICS LABS, 1960.

[39] M. Opper. Learning in neural networks: Solvable dynamics. *EPL (Europhysics Letters)*, 8(4):389, 1989.

[40] D. H. O'Connor, G. M. Wittenberg, and S. S.-H. Wang. Graded bidirectional synaptic plasticity is composed of switch-like unitary events. *Proceedings of the National Academy of Sciences of the United States of America*, 102(27):9679–9684, 2005.

[41] T. M. Bartol, C. Bromer, J. P. Kinney, M. A. Chirillo, J. N. Bourne, K. M. Harris, and T. J. Sejnowski. Hippocampal spine head sizes are highly precise. *bioRxiv*, 2015.

[42] Ö. Türel, J. H. Lee, X. Ma, and K. K. Likharev. Neuromorphic architectures for nanoelectronic circuits. *International Journal of Circuit Theory and Applications*, 32(5):277–302, 2004.

[43] H. Gutfreund and Y. Stein. Capacity of neural networks with discrete synaptic couplings. *Journal of Physics A: Mathematical and General*, 23(12):2613, 1990.

[44] W. Krauth and M. Mézard. Storage capacity of memory networks with binary couplings. *Journal de Physique*, 50(20):3057–3066, 1989.

[45] C. C.H. Petersen, R. C. Malenka, R. A. Nicoll, and J. J. Hopfield. All-or-none potentiation at ca3-ca1 synapses. *Proceedings of the National Academy of Sciences*, 95(8):4732–4737, 1998.

[46] U. S. Bhalla and R. Iyengar. Emergent properties of networks of biological signaling pathways. *Science*, 283(5400):381–387, 1999.

[47] W. Bialek. Stability and noise in biochemical switches. In *Advances in neural information processing systems*, pages 103–109, 2001.

[48] A. Hayer and U. S. Bhalla. Molecular switches at the synapse emerge from receptor and kinase traffic. *PLoS computational biology*, 1(2):e20, 2005.

[49] P. Miller, A. M. Zhabotinsky, J. E. Lisman, and X.-J. Wang. The stability of a stochastic camkii switch: dependence on the number of enzyme molecules and protein turnover. *PLoS biology*, 3(4):e107, 2005.

[50] E. Amaldi. On the complexity of training perceptrons. 1991.

[51] Avrim Blum and Ronald L Rivest. Training a 3-node neural network is np-complete. In *Advances in neural information processing systems*, pages 494–501, 1989.

[52] H. Sompolinsky, N. Tishby, and H. S. Seung. Learning from examples in large neural networks. *Physical Review Letters*, 65(13):1683, 1990.

[53] H. Huang, K.Y. M. Wong, and Y. Kabashima. Entropy landscape of solutions in the binary perceptron problem. *Journal of Physics A: Mathematical and Theoretical*, 46(37):375002, 2013.

[54] T. Obuchi and Y. Kabashima. Weight space structure and analysis using a finite replica number in the ising perceptron. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(12):P12014, 2009.

[55] H. Horner. Dynamics of learning for the binary perceptron problem. *Zeitschrift für Physik B Condensed Matter*, 86(2):291–308, 1992.

[56] Haiping Huang and Yoshiyuki Kabashima. Origin of the computational hardness for learning with binary synapses. *Physical Review E*, 90(5):052813, 2014.

[57] A. Braunstein and R. Zecchina. Learning by message passing in networks of discrete synapses. *Physical review letters*, 96(3):030201, 2006.

[58] Carlo Baldassi and Alfredo Braunstein. A max-sum algorithm for training discrete neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(8):P08008, 2015.

[59] C. Baldassi, A. Braunstein, N. Brunel, and R. Zecchina. Efficient supervised learning in networks with binary synapses. *BMC neuroscience*, 8(2):S13, 2007.

[60] C. Baldassi. Generalization learning in a perceptron with binary synapses. *Journal of Statistical Physics*, 136(5):902–916, 2009.

[61] E. Gardner. Maximum storage capacity in neural networks. *EPL (Europhysics Letters)*, 4(4):481, 1987.

[62] Marc Mezard and Andrea Montanari. *Information, physics, and computation*. Oxford University Press, 2009.

[63] A. Montanari, F. Ricci-Tersenghi, and G. Semerjian. Solving constraint satisfaction problems through belief propagation-guided decimation. *arXiv preprint arXiv:0709.1667*, 2007.

[64] Daniel J Amit. *Modeling brain function: The world of attractor neural networks*. Cambridge university press, 1992.

[65] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[66] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[67] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530, 1985.

[68] Bernard Derrida, Elizabeth Gardner, and Anne Zippelius. An exactly solvable asymmetric neural network model. *EPL (Europhysics Letters)*, 4(2):167, 1987.

[69] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Information storage in neural networks with low levels of activity. *Physical Review A*, 35(5):2293, 1987.

[70] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[71] H Chau Nguyen, Riccardo Zecchina, and Johannes Berg. Inverse statistical problems: from the inverse ising problem to data science. *Advances in Physics*, 66(3):197–261, 2017.

[72] Geoffrey E Hinton and Terrence J Sejnowski. Learning and releaming in boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(282-317):2, 1986.

[73] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre GR Day, Clint Richardson, Charles K Fisher, and David J Schwab. A high-bias, low-variance introduction to machine learning for physicists. *arXiv preprint arXiv:1803.08823*, 2018.

[74] Ruslan Salakhutdinov. Learning and evaluating boltzmann machines. *Tech. Rep., Technical Report UTML TR 2008-002, Department of Computer Science, University of Toronto*, 2008.

[75] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[76] Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pages 599–619. Springer, 2012.

[77] Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. In *Aistats*, volume 10, pages 33–40. Citeseer, 2005.

[78] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses. *Physical review letters*, 115(12):128101, 2015.

[79] Carlo Baldassi, Federica Gerace, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Learning may need only a few bits of synaptic precision. *Physical Review E*, 93(5):052313, 2016.

[80] Piergiorgio Strata and Robin Harvey. Dale's principle. *Brain research bulletin*, 50(5-6):349–350, 1999.

[81] Eleonora Catsigeras. Dale's principle is necessary for an optimal neuronal network's dynamics. *arXiv preprint arXiv:1307.0597*, 2013.

[82] Nicolas Brunel, Vincent Hakim, Philippe Isope, Jean-Pierre Nadal, and Boris Barbour. Optimal information storage and the distribution of synaptic weights: perceptron versus purkinje cell. *Neuron*, 43(5):745–757, 2004.

[83] Bruno A Olshausen and David J Field. Sparse coding of sensory inputs. *Current opinion in neurobiology*, 14(4):481–487, 2004.

[84] Silvio Franz and Giorgio Parisi. Recipes for metastable states in spin glasses. *Journal de Physique I*, 5(11):1401–1415, 1995.

[85] Silvio Franz and Giorgio Parisi. Phase diagram of coupled glassy systems: A mean-field study. *Physical review letters*, 79(13):2486, 1997.

[86] Silvio Franz and Giorgio Parisi. Effective potential in glassy systems: theory and simulations. *Physica A: Statistical Mechanics and its Applications*, 261(3-4):317–339, 1998.

[87] Florent Krzakała, Andrea Montanari, Federico Ricci-Tersenghi, Guilhem Semerjian, and Lenka Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences*, 104(25):10318–10323, 2007.

[88] Carlo Baldassi, Christian Borgs, Jennifer T Chayes, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes. *Proceedings of the National Academy of Sciences*, 113(48):E7655–E7662, 2016.

[89] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Local entropy as a measure for sampling solutions in constraint satisfaction problems. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(2):023301, 2016.

[90] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[91] ET Rolls and G Deco. The noisy brain. *Stochastic dynamics as a principle of brain function.(Oxford Univ. Press, UK, 2010)*, 2010.

[92] Monica Hoyos Flight. Synaptic transmission: On the probability of release. *Nature Reviews Neuroscience*, 9(10):736, 2008.

[93] H Sebastian Seung. Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron*, 40(6):1063–1073, 2003.

[94] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[95] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[96] Carlo Baldassi and Riccardo Zecchina. Efficiency of quantum vs. classical annealing in nonconvex learning problems. *Proceedings of the National Academy of Sciences*, page 201711456, 2018.

[97] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[98] Yonatan Loewenstein and H Sebastian Seung. Operant matching is a generic outcome of synaptic plasticity based on the covariance between reward and neural activity. *Proceedings of the National Academy of Sciences*, 103(41):15224–15229, 2006.

[99] Carlo Baldassi, Federica Gerace, Hilbert J Kappen, Carlo Lucibello, Luca Saglietti, Enzo Tartaglione, and Riccardo Zecchina. On the role of synaptic stochasticity in training low-precision neural networks. *arXiv preprint arXiv:1710.09825*, 2017.

[100] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, pages 1058–1066, 2013.

[101] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[102] Albert Compte, Nicolas Brunel, Patricia S Goldman-Rakic, and Xiao-Jing Wang. Synaptic mechanisms and network dynamics underlying spatial working memory in a cortical network model. *Cerebral Cortex*, 10(9):910–923, 2000.

[103] Kechen Zhang. Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: a theory. *Journal of Neuroscience*, 16(6):2112–2126, 1996.

[104] John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the national academy of sciences*, 81(10):3088–3092, 1984.

[105] Daniel J Amit and Nicolas Brunel. Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cerebral cortex (New York, NY: 1991)*, 7(3):237–252, 1997.

[106] Rony Azouz and Charles M Gray. Cellular mechanisms contributing to response variability of cortical neurons in vivo. *Journal of Neuroscience*, 19(6):2209–2223, 1999.

[107] Robert C Cannon, Cian O'Donnell, and Matthew F Nolan. Stochastic ion channel gating in dendritic neurons: morphology dependence and probabilistic synaptic activation of dendritic spikes. *PLoS computational biology*, 6(8):e1000886, 2010.

[108] Jan W Brascamp, Raymond Van Ee, Andre J Noest, Richard HAH Jacobs, and Albert V van den Berg. The time course of binocular rivalry reveals a fundamental role of noise. *Journal of vision*, 6(11):8–8, 2006.

[109] Lars Buesing, Johannes Bill, Bernhard Nessler, and Wolfgang Maass. Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS computational biology*, 7(11):e1002211, 2011.

[110] HJ Kappen and JJ Spanjers. Mean field theory for asymmetric neural networks. *Physical Review E*, 61(5):5658, 2000.

[111] Yasser Roudi and John Hertz. Mean field theory for nonequilibrium network reconstruction. *Physical review letters*, 106(4):048702, 2011.

[112] M Mézard and J Sakellariou. Exact mean-field inference in asymmetric kinetic ising systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(07):L07001, 2011.

[113] Erik Aurell and Magnus Ekeberg. Inverse ising inference using all the data. *Physical review letters*, 108(9):090201, 2012.

[114] Alfredo Braunstein, Abolfazl Ramezanpour, Riccardo Zecchina, and Pan Zhang. Inference and learning in sparse systems with multiple states. *Physical Review E*, 83(5):056114, 2011.

[115] Chao Huang, Andrey Resnik, Tansu Celikel, and Bernhard Englitz. Adaptive spike threshold enables robust and temporally precise neuronal encoding. *PLoS computational biology*, 12(6):e1004984, 2016.

[116] Alireza Alemi, Carlo Baldassi, Nicolas Brunel, and Riccardo Zecchina. A three-threshold learning rule approaches the maximal capacity of recurrent neural networks. *PLoS computational biology*, 11(8):e1004439, 2015.

[117] Jonathan Binas, Ueli Rutishauser, Giacomo Indiveri, and Michael Pfeiffer. Learning and stabilization of winner-take-all dynamics through interacting excitatory and inhibitory plasticity. *Frontiers in computational neuroscience*, 8:68, 2014.

[118] Rodney J Douglas, Kevan AC Martin, and David Whitteridge. A canonical microcircuit for neocortex. *Neural computation*, 1(4):480–488, 1989.

[119] Vernon B Mountcastle. The columnar organization of the neocortex. *Brain: a journal of neurology*, 120(4):701–722, 1997.

[120] Tom Binzegger, Rodney J Douglas, and Kevan AC Martin. A quantitative map of the circuit of cat primary visual cortex. *Journal of Neuroscience*, 24(39):8441–8453, 2004.

[121] Rodney J Douglas and Kevan AC Martin. Recurrent neuronal circuits in the neocortex. *Current biology*, 17(13):R496–R500, 2007.

[122] Matteo Carandini and David J Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51, 2012.

[123] Sebastian Handrich, Andreas Herzog, Andreas Wolf, and Christoph S Herrmann. A biologically plausible winner-takes-all architecture. In *International Conference on Intelligent Computing*, pages 315–326. Springer, 2009.

[124] Zhi-Hong Mao and Steve G Massaquoi. Dynamics of winner-take-all competition in recurrent neural networks with lateral inhibition. *IEEE transactions on neural networks*, 18(1):55–69, 2007.

[125] Nancy Lynch, Cameron Musco, and Merav Parter. Computational tradeoffs in biological neural networks: Self-stabilizing winner-take-all networks. *arXiv preprint arXiv:1610.02084*, 2016.

[126] Matthias Oster and Shih-Chii Liu. Spiking inputs to a winner-take-all network. In *Advances in Neural Information Processing Systems*, pages 1051–1058, 2006.

[127] Yuguang Fang, Michael A Cohen, and Thomas G Kincaid. Dynamics of a winner-take-all neural network. *Neural Networks*, 9(7):1141–1154, 1996.

[128] Bertrand Fontaine, José Luis Peña, and Romain Brette. Spike-threshold adaptation predicted by membrane potential dynamics in vivo. *PLoS computational biology*, 10(4):e1003560, 2014.

[129] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Statistical mechanics of neural networks near saturation. *Annals of physics*, 173(1):30–67, 1987.

[130] Amos Storkey. Palimpsest memories: a new high-capacity forgetful learning rule for hopfield networks. In *In preparation*. Citeseer, 1998.

[131] Luca Saglietti, Federica Gerace, Alessandro Ingrosso, Carlo Baldassi, and Riccardo Zecchina. From statistical inference to a differential learning rule for stochastic neural networks. *arXiv preprint arXiv:1805.10714*, 2018.

[132] Giorgio Parisi. A memory which forgets. *Journal of Physics A: Mathematical and General*, 19(10):L617, 1986.

[133] David J Thouless, Philip W Anderson, and Robert G Palmer. Solution of'solvable model of a spin glass'. *Philosophical Magazine*, 35(3):593–601, 1977.

[134] Hugo Larochelle, Michael Mandel, Razvan Pascanu, and Yoshua Bengio. Learning algorithms for the classification restricted boltzmann machine. *Journal of Machine Learning Research*, 13(Mar):643–669, 2012.

[135] Peter JM Van Laarhoven and Emile HL Aarts. Simulated annealing. In *Simulated annealing: Theory and applications*, pages 7–15. Springer, 1987.

[136] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

[137] Herbert Jaeger. Controlling recurrent neural networks by conceptors. *arXiv preprint arXiv:1403.3369*, 2014.

[138] LF Abbott, Brian DePasquale, and Raoul-Martin Memmesheimer. Building functional networks of spiking model neurons. *Nature neuroscience*, 19(3):350, 2016.

[139] Brian DePasquale, Mark M Churchland, and LF Abbott. Using firing-rate dynamics to train recurrent networks of spiking model neurons. *arXiv preprint arXiv:1601.07620*, 2016.

[140] R Felix Reinhart and Jochen J Steil. A constrained regularization approach for input-driven recurrent neural networks. *Differential Equations and Dynamical Systems*, 19(1-2):27–46, 2011.

[141] Norbert M Mayer and Matthew Browne. Echo state networks and self-prediction. In *International Workshop on Biologically Inspired Approaches to Advanced Information Technology*, pages 40–48. Springer, 2004.

[142] David Sussillo and LF Abbott. Transferring learning from external to internal weights in echo-state networks with sparse connectivity. *PLoS One*, 7(5):e37372, 2012.

[143] Carlo Baldassi, Federica Gerace, Luca Saglietti, and Riccardo Zecchina. From inverse problems to learning: a statistical mechanics approach. In *Journal of Physics: Conference Series*, volume 955, page 012001. IOP Publishing, 2018.

Appendix A

Belief Propagation (BP)

In Statistical Physics, we typically deal with systems characterized by many components, whose states are defined by mutually dependent random variables, through a series of more or less complex interactions. The scheme of the interactions among random variables is provided by means of graphical models, relying on *factor graphs* [62].

In this regard, suppose to have a system characterized by N interacting random variables $\boldsymbol{x} = \{x_i\}_{i=1}^{N}$ taking values in a finite alphabet χ, whose joint probability distribution can be written in the following factorized form:

$$P(\boldsymbol{x}) = \frac{1}{Z}\prod_{a=1}^{M} P_a(\boldsymbol{x}_{\partial a}), \tag{A.1}$$

where Z is a normalization constant and ∂a denotes a sub-set of the interacting variables, distributed according to $P_a(\boldsymbol{x}_{\partial a})$. For instance, for the 1D Ising model, where $\chi = \{-1, 1\}$, $P(\boldsymbol{x})$ is given by:

$$\begin{aligned} P(\boldsymbol{x}) &= \frac{1}{Z}\exp\left(-\beta H(\boldsymbol{x})\right) \\ &= \frac{1}{Z}\prod_{i=1}^{N-1}\exp\left(\beta J x_i x_{i+1}\right)\prod_{i=1}^{N}\exp\left(\beta h x_i\right), \end{aligned} \tag{A.2}$$

being $H(\boldsymbol{x}) = -J\sum_{i=1}^{N-1} x_i x_{i+1} - h\sum_{i=1}^{N} x_i$, with $J > 0$ the interaction or coupling constant and h the external field. The joint probability distribution can be easily visualized through a graph made of two kinds of nodes:

- the *variable* nodes, associated to the N variables of the problem;
- the *function* nodes, related to the M groups of interacting variables.

In the following, we will use the index i for denoting the variable nodes and the index a for the function nodes.

Belief Propagation is an iterative algorithm that estimates the marginals of the joint probability distribution $P(\boldsymbol{x})$, provides samples according to $P(\boldsymbol{x})$ and computes many interesting quantities arising from $P(\boldsymbol{x})$, such as the partition function or the free energy.

The way it works is the following. It associates to each edge (i,a) of the factor graph, two messages: $\nu^t_{i\to a}(x_i)$, namely the message that propagates from the variable node i to the function node a, and $\hat{\nu}^t_{a\to i}(x_i)$, namely the message that instead propagates in the opposite direction. Here t is the index associated to each iteration of the algorithm. The messages are then updated according to the following BP-equations:

$$\begin{aligned} \nu^{t+1}_{i\to a}(x_i) &\simeq \prod_{b\in\partial i\setminus a} \hat{\nu}^t_{b\to i}(x_i) \\ \hat{\nu}^t_{a\to i}(x_i) &\simeq \sum_{\boldsymbol{x}_{\partial a\setminus i}} P_a(\boldsymbol{x}_{\partial a}) \prod_{j\in\partial a\setminus i} \nu^t_{j\to a}(x_j)\,. \end{aligned} \tag{A.3}$$

These two messages have a precise meaning. Indeed, when $t\to\infty$, $\nu^{\infty}_{i\to a}(x_i)$ simply coincides with the marginal probability distribution of x_i in the specific case the function node a has been removed from the factor graph. Equivalently, $\hat{\nu}^{\infty}_{a\to i}(x_i)$ represents the marginal probability distribution of the variable x_i in absence of the variable node i. The marginal probability distribution is then estimated at the t-th iteration as:

$$\nu_i^t(x_i) \simeq \prod_{a \in \partial i} \hat{\nu}_{a \to i}^{t-1}(x_i). \tag{A.4}$$

This approximation can be proven to be exact for tree factor graphs. However, being the BP equations based on local updates, it can be considered as a valuable estimate even in those regions of loopy factor graphs, where a local tree structure can be detected.

The BP equation in the Discrete Perceptron model

The Belief Propagation algorithm is usually widely exploited in Constraint Satisfaction Problems. In these kinds of problems, a set of random variables has to satisfy a certain number of constraints. This is the case of the classification problem in the discrete Perceptron model, where the constraints are denoted by the set of p input patterns $\{\xi^\mu\}_{\mu=1}^p$, that the network has to properly classify, whereas the microscopic variables are exactly represented by the synaptic couplings $\{W_i\}_{i=1}^N$.

As seen in the main text, the joint probability distribution of the synaptic couplings is given by selecting all those configurations that manage to satisfy all the constraints at once:

$$P(\boldsymbol{W}) = \frac{1}{Z} \prod_{\mu=1}^{p} \Theta\left(y^\mu \left(\frac{\boldsymbol{W} \cdot \boldsymbol{\xi}^\mu}{\sqrt{N}} - \theta\sqrt{N} \right) \right), \tag{A.5}$$

where y^μ defines the class-label associated to the pattern $\boldsymbol{\xi}^\mu$, that is multiplied with the actual response of the network, compared to a local threshold $\theta\sqrt{N}$. The joint probability distribution of the synaptic couplings shows the same factorized form of the probability distribution in Eq. (A.1), where the interactions among variables are here restated in terms of constraints.

In the following, we derive the BP-equations associated to this constraint satisfaction problem, in order to extract the marginals of the joint probability distribution $P(\boldsymbol{W})$. The BP-equations of a discrete Perceptron learning problem can be written in this case as:

$$\begin{aligned} \nu_{i\to\mu}^{t+1}(W_i) &\simeq \prod_{\mu'\in\partial i\setminus\mu} \hat{\nu}_{\mu'\to i}^{t}(W_i) \\ \hat{\nu}_{\mu\to i}^{t}(W_i) &\simeq \sum_{\boldsymbol{W}_{\partial\mu\setminus i}} \Theta\left(y^{\mu}\left(\frac{\boldsymbol{W}\cdot\boldsymbol{\xi}^{\mu}}{\sqrt{N}}-\theta\sqrt{N}\right)\right) \prod_{j\in\partial\mu\setminus i} \nu_{j\to\mu}^{t}\left(W_j\right). \end{aligned} \tag{A.6}$$

We proceed computing the message from function to variable nodes, defined in the last equation. Indeed, the expression of the message from variable to function nodes, directly follows from it. To this end, we can start introducing the following definition:

$$u^{\mu} = \frac{\boldsymbol{W}\cdot\boldsymbol{\xi}^{\mu}}{\sqrt{N}} - \theta\sqrt{N} \tag{A.7}$$

by means of the Dirac-Delta, namely

$$\begin{aligned} \hat{\nu}_{\mu\to i}^{t}(W_i) \simeq \int du^{\mu} \sum_{\boldsymbol{W}_{\partial\mu\setminus j}} \Theta\left(y^{\mu}u^{\mu}\right)\delta\left(u^{\mu}-\frac{\boldsymbol{W}\cdot\boldsymbol{\xi}^{\mu}}{\sqrt{N}}+\theta\sqrt{N}\right) \\ \times \prod_{j\in\partial\mu\setminus i} \nu_{j\to\mu}^{t}\left(W_j\right). \end{aligned} \tag{A.8}$$

We then express the Dirac-Delta through its integral representation:

$$\begin{aligned} \hat{\nu}_{\mu\to i}^{t}(W_i) \simeq \int \frac{du^{\mu}d\hat{u}^{\mu}}{2\pi} \sum_{\boldsymbol{W}_{\partial\mu\setminus i}} \Theta\left(y^{\mu}u^{\mu}\right)\exp\left(i\hat{u}^{\mu}\left(u^{\mu}+\theta\sqrt{N}\right)\right) \\ \times \prod_{j\in\partial\mu\setminus i} \exp\left(-i\hat{u}^{\mu}\frac{W_j\cdot\xi_j^{\mu}}{\sqrt{N}}\right) \prod_{j\in\partial\mu\setminus i} \nu_{j\to\mu}^{t}\left(W_j\right), \end{aligned} \tag{A.9}$$

where we have further exploited the identity $\boldsymbol{W}\cdot\boldsymbol{\xi}^{\mu} = \sum_{j\in\partial\mu\setminus i} W_j\cdot\xi_j^{\mu}$, being the summation over all possible configurations of $\boldsymbol{W}$ restricted only to the nearest

neighbours of the function node μ, except i. Thanks to that, we are now able to factorize over the index j, thus exploiting the following identity:

$$\sum_{\boldsymbol{W}_{\partial\mu\setminus i}} \cdot = \prod_{j\in\partial\mu\setminus i} \int d\mu\left(W_j\right)\cdot, \tag{A.10}$$

where $d\mu\left(W\right)$ denotes the measure over the synaptic couplings, namely

$$d\mu\left(W\right) = \int dW \sum_{l=0}^{L} \delta\left(W-l\right), \tag{A.11}$$

being $L+1$ the total number of available synaptic states. The function to variable message is then given by:

$$\begin{aligned}\hat{v}^{t}_{\mu\to i}\left(W_i\right) \simeq & \int \frac{du^{\mu}d\hat{u}^{\mu}}{2\pi}\Theta\left(y^{\mu}u^{\mu}\right)\exp\left(i\hat{u}^{\mu}\left(u^{\mu}+\theta\sqrt{N}\right)\right) \\ & \times \prod_{j\in\partial\mu\setminus i}\left[\int d\mu\left(W_j\right)\exp\left(-i\hat{u}^{\mu}\frac{W_j\cdot\xi^{\mu}_j}{\sqrt{N}}\right)v^{t}_{j\to\mu}\left(W_j\right)\right].\end{aligned} \tag{A.12}$$

We can now perform an asymptotic expansion of the exponential for large N:

$$\exp\left(-i\hat{u}^{\mu}\frac{W_j\cdot\xi^{\mu}_j}{\sqrt{N}}\right) \simeq 1 - i\hat{u}^{\mu}\frac{W_j\cdot\xi^{\mu}_j}{\sqrt{N}} - \frac{(\hat{u}^{\mu})^2}{2}\frac{\left(W_j\cdot\xi^{\mu}_j\right)^2}{N}. \tag{A.13}$$

Plugging this expansion in Eq. (A.12), we get:

$$\hat{\nu}^t_{\mu\to i}(W_i) \simeq \int \frac{du^\mu d\hat{u}^\mu}{2\pi}\Theta(y^\mu u^\mu)\exp\left(i\hat{u}^\mu\left(u^\mu+\theta\sqrt{N}\right)\right)$$
$$\times \prod_{j\in\partial\mu\setminus i}\left[\int d\mu\left(W_j\right)\nu^t_{j\to\mu}\left(W_j\right)\right.$$
$$\left.\times\exp\left(\log\left(1-i\hat{u}^\mu\frac{W_j\cdot\xi^\mu_j}{\sqrt{N}}-\frac{(\hat{u}^\mu)^2}{2}\frac{\left(W_j\cdot\xi^\mu_j\right)^2}{N}\right)\right)\right], \tag{A.14}$$

where we have further exploited the properties of exponential and logarithmic functions. The variable to function node message, namely $\nu^t_{j\to\mu}\left(W_j\right)$, represents the marginal distribution of the synaptic coupling W_j in absence of the function node μ, at the t-th iteration of the BP equations. Therefore we can write:

$$\hat{\nu}^t_{\mu\to i}(W_i) \simeq \int \frac{du^\mu d\hat{u}^\mu}{2\pi}\Theta(y^\mu u^\mu)\prod_{j\in\partial\mu\setminus i}\left[\exp\left(i\hat{u}^\mu\left(u^\mu+\theta\sqrt{N}\right)\right)\right.$$
$$\left.\times\exp\left(\log\left(1-i\hat{u}^\mu\frac{\langle W_j\rangle\cdot\xi^\mu_j}{\sqrt{N}}-\frac{(\hat{u}^\mu)^2}{2}\frac{\left(\langle W_j\rangle\cdot\xi^\mu_j\right)^2}{N}\right)\right)\right], \tag{A.15}$$

where we have defined: $\langle\cdot\rangle = \int d\mu\left(W_j\right)\cdot\nu^t_{j\to\mu}\left(W_j\right)$. At this point, we can further expand the logarithm for large N, thus obtaining:

$$\hat{\nu}^t_{\mu\to i}(W_i) \simeq \int \frac{du^\mu d\hat{u}^\mu}{2\pi}\Theta(y^\mu u^\mu)\exp\left(i\hat{u}^\mu\left(u^\mu+\theta\sqrt{N}\right)\right)$$
$$\times\prod_{j\in\partial\mu\setminus i}\left[\exp\left(-i\hat{u}^\mu\frac{\langle W_j\rangle\cdot\xi^\mu_j}{\sqrt{N}}-\frac{(\hat{u}^\mu)^2}{2N}\mathrm{Var}\left(W_j\right)\left(\xi^\mu_j\right)^2\right)\right], \tag{A.16}$$

with $\mathrm{Var}\left(W_j\right) = \left\langle W_j^2 \right\rangle - \left\langle W_j \right\rangle^2$. Exploiting the properties of the exponential function, we then obtain:

$$\begin{aligned} \hat{v}^t_{\mu\to i}(W_i) \simeq & \int \frac{du^\mu d\hat{u}^\mu}{2\pi} \Theta\left(y^\mu u^\mu\right) \\ & \times \exp\left(i\hat{u}^\mu \left(u^\mu + \theta\sqrt{N} - \frac{m_W^{\mathrm{cav}}}{\sqrt{N}} \right) - \frac{(\hat{u}^\mu)^2}{2} \frac{\sigma_W^{\mathrm{cav}}}{N} \right), \end{aligned} \tag{A.17}$$

where we have defined:

$$\begin{aligned} m_W^{\mathrm{cav}} &= \sum_j \left\langle W_j \right\rangle \cdot \xi_j^\mu - \left\langle W_i \right\rangle \cdot \xi_i^\mu \\ \sigma_W^{\mathrm{cav}} &= \sum_j \mathrm{Var}\left(W_j\right) \left(\xi_j^\mu\right)^2 - \mathrm{Var}\left(W_i\right) \left(\xi_i^\mu\right)^2. \end{aligned} \tag{A.18}$$

These two quantities represent respectively the cavity mean and variance, since they quantify the mean and the variance of the synaptic couplings distribution, in absence of the synaptic variable node i. We can notice that the $\hat{u}^\mu$-integral is a Gaussian integral, thus it can be easily computed analytically, leading to the following result:

$$\begin{aligned} \hat{v}^t_{\mu\to i}(W_i) \simeq & \int \frac{du^\mu}{\sqrt{2\pi\left(\left(\sigma_W^{\mathrm{cav}}\right)^2/N\right)}} \\ & \times \Theta\left(y^\mu u^\mu\right) \exp\left(- \frac{\left(u^\mu + \theta\sqrt{N} - \frac{m_W^{\mathrm{cav}}}{\sqrt{N}} \right)^2}{2\frac{\left(\sigma_W^{\mathrm{cav}}\right)^2}{N}} \right). \end{aligned} \tag{A.19}$$

The remaining u^μ-integral can be computed analytically as well, by splitting the domain of integration into different intervals, according to the theta-function. We thus finally obtain:

$$\hat{v}^t_{\mu\to i}(W_i) \simeq \begin{cases} \frac{1}{2}\text{erfc}\left(\frac{N\theta - m_W^{\text{cav}}}{\sqrt{2\left(\sigma_W^{\text{cav}}\right)^2}}\right) & \text{if } y^\mu = 1 \\ \\ 1 - \frac{1}{2}\text{erfc}\left(\frac{N\theta - m_W^{\text{cav}}}{\sqrt{2\left(\sigma_W^{\text{cav}}\right)^2}}\right) & \text{if } y^\mu = -1. \end{cases} \tag{A.20}$$

In the end, the BP-equations for a discrete Perceptron model are given by:

$$\begin{aligned} v^{t+1}_{i\to\mu}(W_i) \simeq \prod_{\mu'\in\partial i\setminus\mu} \Bigg[& \frac{1}{2}\text{erfc}\left(\frac{N\theta - m_W^{\text{cav}}}{\sqrt{2\left(\sigma_W^{\text{cav}}\right)^2}}\right)\delta\left(y^{\mu'} - 1\right) + \\ & + \left(1 - \frac{1}{2}\text{erfc}\left(\frac{N\theta - m_W^{\text{cav}}}{\sqrt{2\left(\sigma_W^{\text{cav}}\right)^2}}\right)\right)\delta\left(y^{\mu'} + 1\right)\Bigg] \\ \\ \hat{v}^t_{\mu\to i}(W_i) \simeq \Bigg[& \frac{1}{2}\text{erfc}\left(\frac{N\theta - m_W^{\text{cav}}}{\sqrt{2\left(\sigma_W^{\text{cav}}\right)^2}}\right)\delta\left(y^{\mu} - 1\right) + \\ & + \left(1 - \frac{1}{2}\text{erfc}\left(\frac{N\theta - m_W^{\text{cav}}}{\sqrt{2\left(\sigma_W^{\text{cav}}\right)^2}}\right)\right)\delta\left(y^{\mu} + 1\right)\Bigg]. \end{aligned} \tag{A.21}$$

These equations need to be iterated till a fixed point is reached. From the messages, as we said, we can then compute every other interesting quantity, such as the partition function, the free energy and, in particular, the entropy.

Appendix B

Statistical Physics Analysis of the Stochastic Perceptron

B.1 Energy of a Binarized Configuration

In this section, we compute the average training error associated to a binarized configuration, i.e. $W_i = \text{sign}(m_i)$, namely

$$E\left(q_*\right) = 1 - \lim_{N\to\infty} \frac{1}{\alpha N} \mathbb{E}_{P(\boldsymbol{\xi}^\mu, y^\mu)} \left[\sum_{\mu=1}^{p} \left\langle \Theta\left(y^\mu \sum_i \text{sign}(m_i) \xi_i^\mu \right) \right\rangle \right], \tag{B.1}$$

whit $\langle\cdot\rangle$ being the thermal average, defined as:

$$\langle\cdot\rangle = \frac{\int \mathscr{D}m \cdot \delta\left(\sum_i m_i^2 - q_* N\right) \, \exp\left(\beta \mathscr{L}(m)\right)}{\int \mathscr{D}m \, \delta\left(\sum_i m_i^2 - q_* N\right) \, \exp\left(\beta \mathscr{L}(m)\right)}. \tag{B.2}$$

In the specific case of random identically and independently distributed input data $\left\{\boldsymbol{\xi}^\mu, y^\mu\right\}_{\mu=1}^{p}$, we can simply neglect the summation over all patterns and then compute the resulting mean training error per pattern:

$$e\left(q_*\right) = 1 - \lim_{N\to\infty} \frac{1}{\alpha N} \mathbb{E}_{P(\boldsymbol{\xi}^\mu, y^\mu)} \left\langle \Theta\left(y^\mu \sum_i \text{sign}(m_i) \xi_i^\mu \right) \right\rangle. \tag{B.3}$$

The quenched average can be tackled by means of the replica formalism. To this end, we imagine to replicate our system n times, so that we can write:

$$\begin{aligned}
&\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left\langle \Theta\left(y^{\mu}\sum_{i}\operatorname{sign}(m_i)\xi_i^{\mu}\right)\right\rangle = \\
&= \lim_{n\to 0}\mathbb{E}_{\xi}\left[\int\prod_{a=1}^{n}\mathscr{D}m^a\,\delta\left(\sum_{i}(m_i^a)^2 - q_{aa}N\right)\theta\left(\frac{\sum_i \operatorname{sign}(m_i^1)\xi_i^1}{\sqrt{N}}\right)\right. \\
&\left.\times\prod_{\mu,a}H^{\beta}\left(-\frac{\sum_i \xi_i^{\mu}m_i^a}{\sqrt{N}\left(\sqrt{1-q_*}\right)}\right)\right],
\end{aligned} \tag{B.4}$$

where we have made explicit the expression of the log-likelihood $\mathscr{L}(m^a)$ in Eq. (3.7). We can then introduce the following auxiliary variables:

$$\begin{aligned}
u_a^{\mu} &= \frac{1}{\sqrt{N}}\sum_i \xi_i^{\mu}m_i^a \\
\tilde{u} &= \frac{1}{\sqrt{N}}\sum_i \operatorname{sign}(m_i^1)\xi_i^1
\end{aligned} \tag{B.5}$$

through Dirac-deltas and then exploit their integral representation, so that in the end we obtain:

$$\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left\langle \Theta\left(y^{\mu}\sum_{i}\text{sign}(m_i)\xi_i^{\mu}\right)\right\rangle =$$

$$= \lim_{n\to 0}\int\prod_{i,a}dm_i^a\int\prod_{a}\frac{d\hat{q}_{aa}}{2\pi}\exp\left(-\frac{1}{2}\sum_{a}\left(N\hat{q}_{aa}q_{aa}-\hat{q}_{aa}\sum_{i}(m_i^a)^2\right)\right)$$

$$\times\int\prod_{\mu,a}\frac{du_a^{\mu}d\hat{u}_a^{\mu}}{2\pi}\prod_{\mu,a}H^{\beta}\left(-\frac{u_a^{\mu}}{\sqrt{1-q_*}}\right)\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left[\int\frac{d\tilde{u}d\hat{\tilde{u}}}{2\pi}\,\theta(\tilde{u})\right. \tag{B.6}$$

$$\left.\times\exp\left(i\sum_{a,\mu}u_a^{\mu}\hat{u}_a^{\mu}-i\sum_{\mu,a,i}\frac{\xi_i^{\mu}m_i^a}{\sqrt{N}}\hat{u}_a^{\mu}+i\tilde{u}\hat{\tilde{u}}-i\frac{\sum_i\text{sign}(m_i^1)\xi_i^1}{\sqrt{N}}\hat{\tilde{u}}\right)\right].$$

The average over the training set can be carried out, by assuming that: $\bar{\xi}_i^{\mu}=0$ and $\text{Var}\left[\xi_i^{\mu}\right]=1$. In order to compute this average, we need however to distinguish between two different cases:

- $\mu\neq 1$:

$$\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left[\exp\left(-\frac{i}{\sqrt{N}}\sum_{a,i}\xi_i^{\mu}m_i^a\hat{u}_a^{\mu}\right)\right]=\exp\left(-\frac{1}{2N}\sum_{i}\sum_{a,b}m_i^am_i^b\hat{u}_a^{\mu}\hat{u}_b^{\mu}\right)$$

- $\mu=1$:

$$\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left[\exp\left(-\frac{i}{\sqrt{N}}\sum_{i}\left(\sum_{a}m_i^a\hat{u}_a^1+\text{sign}(m_i^1)\hat{\tilde{u}}\right)\right)\right]=$$

$$=\exp\left(-\frac{1}{2N}\sum_{i}\left(\sum_{a,b}m_i^am_i^b\hat{u}_a^1\hat{u}_b^1+\hat{\tilde{u}}^2+2\text{sign}(m_i^1)\hat{\tilde{u}}\sum_{a}m_i^a\hat{u}_a^1\right)\right).$$

Recombining the two cases into one single expression, we then obtain:

$$\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left[\theta\left(\frac{\sum_i \text{sign}(m_i)\xi_i^{\mu}}{\sqrt{N}}\right)\right] =$$

$$= \lim_{n\to 0}\int\prod_{i,a} dm_i^a \int\prod_a \frac{d\hat{q}_{aa}}{2\pi}\prod_a \exp\left(-\frac{1}{2}\sum_a N\hat{q}_{aa}q_{aa} - \hat{q}_{aa}\sum_i (m_i^a)^2\right)$$

$$\times\int\prod_{\mu,a}\frac{du_a^{\mu}d\hat{u}_a^{\mu}}{2\pi}\prod_{\mu,a} H^{\beta}\left(-\frac{u_a^{\mu}}{\sqrt{1-q_*}}\right)\times\int\frac{d\tilde{u}d\hat{\tilde{u}}}{2\pi}\,\theta(\tilde{u})\exp\left(i\sum_{a,\mu}\hat{u}_a^{\mu}u_a^{\mu} + i\hat{\tilde{u}}\tilde{u}\right)$$

$$\times\exp\left(-\frac{1}{2N}\sum_{\mu,i}\sum_{a,b}\hat{u}_a^{\mu}\hat{u}_b^{\mu}m_i^a m_i^b - \frac{1}{2N}\sum_i\left(\hat{\tilde{u}}^2 + 2\text{sign}(m_i^1)\hat{\tilde{u}}\sum_a \hat{u}_a^1 m_i^a\right)\right). \tag{B.7}$$

At this point, we introduce the overlap parameters, namely

$$\begin{aligned} q_{ab} &= \frac{1}{N}\sum_i m_i^a m_i^b \\ p_a &= \frac{1}{N}\sum_i \text{sign}(m_i^1)m_i^a \end{aligned} \tag{B.8}$$

through Dirac-deltas, that can be then expressed through their integral representation. Therefore, we obtain:

$$\mathbb{E}_{P(\boldsymbol{\xi}^{\mu},y^{\mu})}\left[\theta\left(\frac{\sum_i \mathrm{sign}(m_i)\xi_i^{\mu}}{\sqrt{N}}\right)\right] =$$

$$= \lim_{n\to 0}\int \prod_{a<b} dq_{ab} \prod_{a\leq b}\frac{d\hat{q}_{ab}}{2\pi}\int\prod_a \frac{dp_a d\hat{p}_a}{2\pi}\exp\left(-N\left(\frac{1}{2}\sum_{a,b}\hat{q}_{ab}q_{ab}+\sum_a \hat{p}_a p_a\right)\right)$$

$$\times \mathscr{G}_S^N\left(\hat{q}_*,\hat{q},\hat{p},\hat{\tilde{p}}\right)\times \mathscr{G}_E^{M-1}\left(q_*,q\right)\times \mathscr{G}_E'\left(q_*,q,p,\tilde{p}\right), \tag{B.9}$$

where we have defined:

$$\mathscr{G}_S = \int_{-1}^{1}\prod_a dm^a \exp\left(\frac{1}{2}\sum_{a,b}\hat{q}_{ab}m^a m^b + \mathrm{sign}(m^1)\sum_a \hat{p}_a m^a\right)$$

$$\mathscr{G}_E = \int\prod_a \frac{du_a d\hat{u}_a}{2\pi}\prod_a H^{\beta}\left(-\frac{u_a}{\sqrt{1-q_*}}\right)\exp\left(i\sum_a u_a\hat{u}_a - \frac{1}{2}\sum_{a,b}q_{ab}\hat{u}_a\hat{u}_b\right)$$

$$\mathscr{G}_E' = \int\prod_a \frac{du_a d\hat{u}_a}{2\pi}\int\frac{d\tilde{u}d\hat{\tilde{u}}}{2\pi}\prod_a H^{\beta}\left(-\frac{u_a}{\sqrt{1-q_*}}\right)\theta(\tilde{u})$$
$$\times\exp\left(i\sum_a u_a\hat{u}_a - \frac{1}{2}\hat{\tilde{u}}^2 + i\tilde{u}\hat{\tilde{u}} - \frac{1}{2}\sum_{a,b}q_{ab}\hat{u}_a\hat{u}_b - \hat{\tilde{u}}\sum_a p_a\hat{u}_a\right) \tag{B.10}$$

The two factors $\mathscr{G}_S$ and $\mathscr{G}_E$ are respectively called *entropic* and *energetic* part. The term entropic refers to the fact that $\mathscr{G}_S$ is defined through a sum over all synaptic couplings satisfying a given constraint, while the term energetic refers to the fact that $\mathscr{G}_E$ contains information on the energy function of the model.

At this point, we apply the RS assumption, even if we need to break the symmetry for making a distinction between the order parameter q_{ab}, describing the overlap between two distinct replicas, and the self-overlap q_{aa}, constrained to the mean

squared norm q_*. Moreover, we have to make a further distinction between the cases $a = 1$ and $a \neq 1$, associated to the p_a overlap parameter. Then:

$$q_{ab} = \begin{cases} q_* & \text{if } a = b \\ q & \text{if } a \neq b \end{cases}$$
$$p_a = \begin{cases} p & \text{if } a = 1 \\ \tilde{p} & \text{if } a \neq 1 \end{cases}. \tag{B.11}$$

The same holds for the conjugate parameters. Applying the RS assumption on the structure of the order parameters, we obtain, for the entropic part:

$$\mathscr{G}_S = \int \mathscr{D}z \int_{-1}^{1} dm^1 \left[\int_{-1}^{1} dm \exp\left(\sqrt{\hat{q}} z m + \frac{1}{2} (\hat{q}_* - \hat{q}) m^2 + \hat{\tilde{p}} \operatorname{sign}(m^1) m \right) \right]^{n-1}, \tag{B.12}$$

while, for the energetic part:

$$\mathscr{G}_E = \int \mathscr{D}z \left[\int \frac{du d\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}} \right) \exp\left(-\frac{1}{2} (q_* - q) \hat{u}^2 + i (\sqrt{q} z + u) \hat{u} \right) \right]^n. \tag{B.13}$$

Finally, for $\mathscr{G}'_E$ we get:

$$\mathscr{G}'_E = \int \mathscr{D}z \int \mathscr{D}\hat{\tilde{u}} \int \frac{d\tilde{u}}{\sqrt{2\pi}} \theta(\tilde{u}) \exp\left(i\tilde{u}\hat{\tilde{u}}\right)$$

$$\times \left[\int \frac{du d\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \exp\left(-\frac{1}{2}(q_* - q)\hat{u}^2 + i\sqrt{q}z\hat{u} + iu\hat{u} - \tilde{p}\hat{u}\hat{\tilde{u}}\right)\right]^{n-1}$$

$$\times \int \frac{du d\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \exp\left(-\frac{1}{2}(q_* - q)\hat{u}^2 + i\sqrt{q}z\hat{u} + iu\hat{u} - p\hat{u}\hat{\tilde{u}}\right). \tag{B.14}$$

Taking the limit of $n \to 0$, we then obtain:

$$\mathscr{G}_S = \int \mathscr{D}z \int_{-1}^{1} dm^1 \frac{\exp\left(\sqrt{\hat{q}}zm^1 + \frac{1}{2}(\hat{q}_* - \hat{q})\left(m^1\right)^2 + \hat{p}\,\mathrm{sign}(m^1)m^1\right)}{\int_{-1}^{1} dm \exp\left(\sqrt{\hat{q}}zm + \frac{1}{2}(\hat{q}_* - \hat{q})m^2 + \hat{\tilde{p}}\,\mathrm{sign}(m^1)m\right)}$$

$$\mathscr{G}_E = 1$$

$$\mathscr{G}'_E = \int \mathscr{D}z \int \mathscr{D}\hat{\tilde{u}} \int \frac{d\tilde{u}}{\sqrt{2\pi}} \theta(\tilde{u}) \exp\left(i\hat{\tilde{u}}\tilde{u}\right)$$

$$\times \left[\int \frac{du d\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \exp\left(-\frac{1}{2}(q_* - q)\hat{u}^2 + iu\hat{u} + i\left(\sqrt{q}z + i\tilde{p}\hat{\tilde{u}}\right)\hat{u}\right)\right]^{-1}$$

$$\times \int \frac{du d\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \exp\left(-\frac{1}{2}(q_* - q)\hat{u}^2 + iu\hat{u} + i\left(\sqrt{q}z + ip\hat{\tilde{u}}\right)\hat{u}\right). \tag{B.15}$$

In order to solve the integrals in $\mathscr{G}'_E$, first of all, we perform the following rotation:

$$
\begin{aligned}
z' &= \frac{\sqrt{q}z + i\tilde{p}\hat{\tilde{u}}}{\sqrt{q-\tilde{p}^2}} \rightarrow z = \frac{\sqrt{q}z' - i\tilde{p}\hat{\tilde{u}}'}{\sqrt{q-\tilde{p}^2}} \\
\hat{\tilde{u}}' &= \frac{\sqrt{\tilde{q}}\hat{\tilde{u}} - i\tilde{p}z}{\sqrt{q-\tilde{p}^2}} \rightarrow \hat{\tilde{u}} = \frac{\sqrt{q}\hat{\tilde{u}}' + i\tilde{p}z'}{\sqrt{q-\tilde{p}^2}}.
\end{aligned}
\tag{B.16}
$$

In this way, we can rewrite $\mathscr{G}'_E$ in the form:

$$
\mathscr{G}'_E(q_*, q, p, \tilde{p}) = \int \mathscr{D}z \frac{f(z)}{g(z)}, \tag{B.17}
$$

where the denominator, performing the $\hat{u}$-Gaussian integral, is given by:

$$
g(z) = \int \mathscr{D}u\, H^{\beta}\left(-\frac{u\sqrt{q_\star - q} - z\sqrt{q-\tilde{p}^2}}{\sqrt{1-q_*}}\right), \tag{B.18}
$$

whereas, the numerator is defined as:

$$
\begin{aligned}
f(z) = \int \mathscr{D}\hat{\tilde{u}} \int \frac{d\tilde{u}}{\sqrt{2\pi}} \theta(\tilde{u}) \exp\left(i\left(\frac{\sqrt{q}\hat{\tilde{u}} + i\tilde{p}z}{\sqrt{q-\tilde{p}^2}}\right)\tilde{u}\right) \int \frac{du d\hat{u}}{2\pi}\, H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \\
\times \exp\left(-\frac{1}{2}(q_* - q)\hat{u}^2 + i\left(u + \frac{(q-p\tilde{p})z + i\sqrt{q}(p-\tilde{p})\hat{\tilde{u}}}{\sqrt{q-\tilde{p}^2}}\right)\hat{u}\right).
\end{aligned}
\tag{B.19}
$$

We now proceed computing the integrals appearing in the numerator $f(z)$, starting from $\hat{\tilde{u}}$-integral. Grouping all together the $\hat{\tilde{u}}$-dependent contributions, we recognize the $\hat{\tilde{u}}$-integral to be a Gaussian integral, thus obtaining:

$$f(z) = \int \frac{dud\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \int \frac{\mathscr{D}\hat{\tilde{u}}d\tilde{u}}{\sqrt{2\pi}} \theta(\tilde{u}) \exp\left(i\left(\frac{\sqrt{q}\tilde{u} + i\sqrt{q}(p-\tilde{p})\hat{u}}{\sqrt{q-\tilde{p}^2}}\right)\hat{\tilde{u}}\right)$$

$$\times \exp\left(-\frac{1}{2}(q_*-q)\hat{u}^2 + i\left(u + \frac{(q-p\tilde{p})z}{\sqrt{q-\tilde{p}^2}}\right)\hat{u} + i\frac{i\tilde{p}z}{\sqrt{q-\tilde{p}^2}}\tilde{u}\right)$$

$$= \int \frac{dud\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \int \frac{d\tilde{u}}{\sqrt{2\pi}} \theta(\tilde{u}) \exp\left(-\frac{1}{2}\left(\frac{\sqrt{q}\tilde{u} + i\sqrt{q}(p-\tilde{p})\hat{u}}{\sqrt{q-\tilde{p}^2}}\right)^2\right)$$

$$\times \exp\left(-\frac{1}{2}(q_*-q)\hat{u}^2 + i\left(u + \frac{(q-p\tilde{p})z}{\sqrt{q-\tilde{p}^2}}\right)\hat{u} + i\frac{i\tilde{p}z}{\sqrt{q-\tilde{p}^2}}\tilde{u}\right).$$

We can then compute the $\tilde{u}$-integral by first expanding and then completing the squares. This makes the $\tilde{u}$-integral to be a Gaussian integral, that we can easily solve:

$$f(z) = \frac{\sqrt{q-\tilde{p}^2}}{\sqrt{q}} \int \frac{dud\hat{u}}{2\pi} H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) H\left(\frac{i\sqrt{q}\left((p-\tilde{p})\hat{u} - \frac{i\tilde{p}\sqrt{q-\tilde{p}^2}}{q}z\right)}{\sqrt{q-\tilde{p}^2}}\right)$$

$$\times \exp\left(-\frac{1}{2}(q_\star - q)\hat{u}^2 + i\hat{u}\left(u + \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}}z\right) + \frac{1}{2}\frac{\tilde{p}^2 z^2}{q}\right). \tag{B.20}$$

The $\hat{u}$-integral, after some change of variables, can be rewritten as:

$$
\begin{aligned}
f(z) = & \frac{\sqrt{q-\tilde{p}^2}}{\sqrt{q(q_*-q)}} \exp\left(\frac{1}{2}\frac{\tilde{p}^2 z^2}{q}\right) \int \frac{du}{\sqrt{2\pi}}\, H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \\
& \times \exp\left(-\frac{1}{2(q_\star - q)}\left(u + \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}} z\right)^2\right) \\
& \times \int \mathscr{D}\hat{u}\, H\left(\frac{\frac{(p-\tilde{p})}{\sqrt{q_*-q}}\hat{u} + i\frac{(p-\tilde{p})}{q_*-q}\left(u + \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}} z\right) - \frac{i\tilde{p}\sqrt{q-\tilde{p}^2}}{q} z}{\frac{\sqrt{q-\tilde{p}^2}}{i\sqrt{q}}}\right).
\end{aligned}
\tag{B.21}
$$

Then, to perform the integration we take advantage of the formula for the integration of error functions, namely

$$
\int \mathscr{D}x\, H\left(\frac{Ex+F}{G}\right) = \frac{F}{\sqrt{E^2+G^2}}. \tag{B.22}
$$

This formula, applied to the $\hat{u}$-integral, finally gives:

$$
\begin{aligned}
f(z) = & \frac{\sqrt{q-\tilde{p}^2}}{\sqrt{q(q_*-q)}} \int \frac{du}{\sqrt{2\pi}}\, H^{\beta}\left(-\frac{u}{\sqrt{1-q_*}}\right) \\
& \times \exp\left(-\frac{1}{2(q_\star - q)}\left(u + \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}} z\right)^2 + \frac{1}{2}\frac{\tilde{p}^2 z^2}{q}\right) \\
& \times H\left(\frac{\left(i\frac{(p-\tilde{p})}{q_*-q}\left(u + \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}} z\right) - \frac{i\tilde{p}\sqrt{q-\tilde{p}^2}}{q} z\right)}{\sqrt{\frac{(p-\tilde{p})^2}{q_*-q} - \frac{q-\tilde{p}^2}{q}}}\right).
\end{aligned}
\tag{B.23}
$$

To simplify this expression we can make the further change of variable $u + \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}} z \to u$, so that we get:

$$f(z) = \frac{\sqrt{q-\tilde{p}^2}}{\sqrt{q}} \exp\left(\frac{1}{2}\frac{\tilde{p}^2 z^2}{q}\right) \int \mathscr{D}u\, H^{\beta}\left(-\frac{\sqrt{q_*-q}u - \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}}z}{\sqrt{1-q_*}}\right) \times H\left(\frac{\left(i\frac{(p-\tilde{p})}{\sqrt{q_*-q}}u - \frac{i\tilde{p}\sqrt{q-\tilde{p}^2}}{q}z\right)}{\sqrt{\frac{(p-\tilde{p})^2}{q_*-q} - \frac{q-\tilde{p}^2}{q}}}\right). \tag{B.24}$$

At this point, we can combine together the numerator $f(z)$ with the denominator $g(z)$, thus obtaining the full expression for $\mathscr{G}'_E$, namely

$$\mathscr{G}'_E = \frac{\sqrt{q-\tilde{p}^2}}{\sqrt{q}} \int \mathscr{D}z \left[\int \mathscr{D}u\, H^{\beta}\left(-\frac{u\sqrt{q_\star - q} - z\sqrt{q-\tilde{p}^2}}{\sqrt{1-q_*}}\right)\right]^{-1} \exp\left(\frac{1}{2}\frac{\tilde{p}^2 z^2}{q}\right) \times \int \mathscr{D}u\, H^{\beta}\left(-\frac{\sqrt{q_*-q}u - \frac{(q-\tilde{p}^2)}{\sqrt{(q-\tilde{p}^2)}}z}{\sqrt{1-q_*}}\right) H\left(\frac{\left(i\frac{(p-\tilde{p})}{\sqrt{q_*-q}}u - \frac{i\tilde{p}\sqrt{q-\tilde{p}^2}}{q}z\right)}{\sqrt{\frac{(p-\tilde{p})^2}{q_*-q} - \frac{q-\tilde{p}^2}{q}}}\right). \tag{B.25}$$

To simplify this expression we can perform the further change of variable $\frac{\tilde{p}}{\sqrt{q}} z \to z$, which in the end gives back the very last expression for $\mathscr{G}'_E$, namely

$$\mathscr{G}'_E = \int \mathscr{D}z \frac{\int \mathscr{D}u\, H\left(-\frac{\left(\frac{(p-\tilde{p})}{\sqrt{q_*-q}}u - \frac{\tilde{p}}{\sqrt{q}}z\right)}{\sqrt{1-\frac{\tilde{p}^2}{q} - \frac{(p-\tilde{p})^2}{q_*-q}}}\right) H^{\beta}\left(-\frac{\sqrt{q_*-q}u - \sqrt{q}z}{\sqrt{1-q_*}}\right)}{\int \mathscr{D}u\, H^{\beta}\left(-\frac{u\sqrt{q_\star - q} - \sqrt{q}z}{\sqrt{1-q_*}}\right)}. \tag{B.26}$$

The remaining integrals in u and z have to be solved numerically, thanks to the help of one of the different methods provided for numerical integration. The mean training error per pattern is finally given by:

$$\begin{aligned} e(q_*) = 1 - \lim_{N\to\infty} \frac{1}{\alpha N} \lim_{n\to 0} \int \prod_{a<b} dq_{ab} \prod_{a\leq b} \frac{d\hat{q}_{ab}}{2\pi} \\ \times \int \prod_a \frac{dp_a d\hat{p}_a}{2\pi} \exp\left(-N\psi\left(q_{ab},\hat{q}_{ab},p_a,\hat{p}_a\right)\right) \mathscr{G}'_E\left(q_{ab},p_a\right), \end{aligned} \tag{B.27}$$

where, within the RS assumption and for $n = 0$, the integrand is given by, for what concerns the action $\psi(x)$:

$$\begin{aligned} \psi &= -\frac{1}{2} n q_* \hat{q}_* - \frac{1}{2} n(n-1) q\hat{q} - p\hat{p} - (n-1)\tilde{p}\hat{\tilde{p}} + \ln \mathscr{G}_S + \alpha \ln \mathscr{G}_E \\ &= -p\hat{p} + \tilde{p}\hat{\tilde{p}} + \ln \mathscr{G}_S + \alpha \ln \mathscr{G}_E, \end{aligned} \tag{B.28}$$

whit the entropic and the energetic part given by:

$$\mathscr{G}_S = \int \mathscr{D}z \int_{-1}^{1} dm^1 \frac{\exp\left(\sqrt{\hat{q}} z m^1 + \frac{1}{2}\left(\hat{q}_* - \hat{q}\right)\left(m^1\right)^2 + \hat{p}\,\mathrm{sign}(m^1) m^1\right)}{\int_{-1}^{1} dm \exp\left(\sqrt{\hat{q}} z m + \frac{1}{2}\left(\hat{q}_* - \hat{q}\right) m^2 + \hat{\tilde{p}}\,\mathrm{sign}(m^1) m\right)} \tag{B.29}$$

$$\mathscr{G}_E = 1,$$

whereas, for what concerns $\mathscr{G}'_E$, it is the one provided in Eq. (B.26). The integrals over the overlap parameters can be solved through the saddle-point approximation, thus leading to the following set of saddle-point equations:

$$\left.\frac{\partial \psi}{\partial p}\right|_{sp} = 0 \rightarrow \hat{p}_{sp} = 0$$

$$\left.\frac{\partial \psi}{\partial \tilde{p}}\right|_{sp} = 0 \rightarrow \hat{\tilde{p}}_{sp} = 0$$

$$\left.\frac{\partial \psi}{\partial \hat{p}}\right|_{sp} = 0 \rightarrow p_{sp} = \int \mathscr{D}z \frac{\int_{-1}^{1} dm\, \mathrm{sign}(m) m \exp\left(-\frac{1}{2}(\hat{q}-\hat{q}_*)m^2 + \sqrt{\hat{q}}zm\right)}{\int_{-1}^{1} dm \exp\left(-\frac{1}{2}(\hat{q}-\hat{q}_*)m^2 + \sqrt{\hat{q}}zm\right)}$$

$$\left.\frac{\partial \psi}{\partial \hat{\tilde{p}}}\right|_{sp} = 0 \rightarrow \tilde{p}_{sp} = \int \mathscr{D}z \frac{\int_{-1}^{1} dm \exp\left(-\frac{1}{2}(\hat{q}-\hat{q}_*)m^2 + \sqrt{\hat{q}}zm\right) \mathrm{sign}(m)}{\left[\int_{-1}^{1} dm \exp\left(-\frac{1}{2}(\hat{q}-\hat{q}_*)m^2 + \sqrt{\hat{q}}zm\right)\right]^2} \times \int_{-1}^{1} dm\, m \exp\left(-\frac{1}{2}(\hat{q}-\hat{q}_*)m^2 + \sqrt{\hat{q}}zm\right), \tag{B.30}$$

where the subscript "sp" is a short for saddle-point. Notice that, at the saddle point: $\mathscr{G}_S = 1$. Within the saddle point approximation, these equations provide the values of the optimal parameters that contribute the most to the integral in Eq. (B.27), thus allowing for a direct estimation of the average training error, associated to a binarized configuration, as a function of the mean squared norm of the control parameters, namely q_*.

Appendix C

DCM learning rule

C.1 DCM derivation

The Delayed Correlation Matching (DCM) learning rule, described in the main text, tries to match the dynamics of the network in presence of an external field of intensity λ_1, described by the transition probability $P(\boldsymbol{s}'|\boldsymbol{s};\lambda_1)$, and the dynamics of the same network but subject to a weaker external field λ_2, described by the transition probability $P(\boldsymbol{s}'|\boldsymbol{s};\lambda_2)$. A measure of distance between two probability distributions is provided by the Kullback-Leibler (KL) divergence. In the specific case we are here considering, we can write the KL divergence as:

$$\langle \mathrm{KL}\left(P(\cdot|\boldsymbol{s};\lambda_1)\,||P(\cdot|\boldsymbol{s};\lambda_2)\right)\rangle_P = \sum_{\{\boldsymbol{s}\}} P(\boldsymbol{s}) \sum_{\{\boldsymbol{s}'\}} P\left(\boldsymbol{s}'|\boldsymbol{s};\lambda_1\right) \log \frac{P(\boldsymbol{s}'|\boldsymbol{s};\lambda_1)}{P(\boldsymbol{s}'|\boldsymbol{s};\lambda_2)}, \quad \text{(C.1)}$$

where we further average over all possible configurations of the initial state $P(s)$. The transition probabilities are defined according to the Glauber dynamics in Eq. (4.1), where the explicit expression of the sigmoidal-shaped activation function depends on the neuronal state model we are considering. In particular, in the case of $s_i \in \{-1,1\}$ state variables, the activation function is given by:

$$\sigma_{\pm 1}\left(s|h^{\lambda};\beta\right) = \frac{e^{-\beta s h^{\lambda}}}{e^{\beta h} + e^{-\beta h^{\lambda}}}. \quad \text{(C.2)}$$

On the contrary, in the case of sparse neuronal models, $s_i \in \{0,1\}$ it can be written as:

$$\sigma_{01}\left(s|h^{\lambda};\beta\right) = \frac{e^{-\beta s h^{\lambda}}}{1+e^{-\beta h^{\lambda}}}, \tag{C.3}$$

where $h_i^{\lambda} = h_i^{\text{ext},\lambda} + \sum_{j\neq i} W_{ij}^{\lambda} s_j - \theta_i^{\lambda}$ is as usual the local field, given by the contribution of both the external field and the recurrent signal from surrounding neurons, while β is the inverse temperature parameter.

Replacing the definition of the Glauber transition probability of Eq. (4.1) in Eq. (C.1), we obtain the more explicit expression for the average KL divergence:

$$\left\langle \mathrm{KL}\left(P\left(\cdot|\boldsymbol{s};\lambda_1\right)||P\left(\cdot|\boldsymbol{s};\lambda_2\right)\right)\right\rangle_P = \sum_{\{\boldsymbol{s}\}} P(\boldsymbol{s}) \sum_i \sum_{s_i'} \sigma\left(s_i'|h_i^{\lambda_1}\right) \log \frac{\sigma\left(s_i'|h_i^{\lambda_1}\right)}{\sigma\left(s_i'|h_i^{\lambda_2}\right)}. \tag{C.4}$$

To determine the configuration of the network parameters, namely synaptic couplings and thresholds, optimizing the KL divergence, we then need to differentiate Eq. (C.4), with respect to $W_{ik}^{\lambda_2}$ and $\theta_i^{\lambda_2}$, thus obtaining:

$$\begin{aligned}
-\frac{\partial}{\partial W_{ik}^{\lambda_2}} \left\langle \mathrm{KL}\left(P\left(\cdot|\boldsymbol{s};\lambda_1\right)||P\left(\cdot|\boldsymbol{s};\lambda_2\right)\right)\right\rangle_P &= \sum_{\{\boldsymbol{s}\}} P(\boldsymbol{s}) \sum_{s_i'} \sigma\left(s_i'|h_i^{\lambda_1}\right)\left(s_i' - \left\langle s_i'\right\rangle\right) s_k \\
&= \left\langle s_i' s_k\right\rangle_{P,\lambda_1} - \left\langle s_i' s_k\right\rangle_{P,\lambda_2} \\
-\frac{\partial}{\partial \theta_i^{\lambda_2}} \left\langle \mathrm{KL}\left(P\left(\cdot|\boldsymbol{s};\lambda_1\right)||P\left(\cdot|\boldsymbol{s};\lambda_2\right)\right)\right\rangle_P &= \sum_{\{\boldsymbol{s}\}} P(\boldsymbol{s}) \sum_{s_i'} \sigma\left(s_i'|h_i^{\lambda_1}\right)\left(s_i' - \left\langle s_i'\right\rangle\right) \\
&= -\left(\left\langle s_i'\right\rangle_{P,\lambda_1} - \left\langle s_i'\right\rangle_{P,\lambda_2}\right),
\end{aligned}$$

where we have exploited the identity:

$$\frac{\partial}{\partial h} \log\sigma\left(s|h;\beta\right) = s - \left\langle s\right\rangle_h, \tag{C.5}$$

with $\left\langle s\right\rangle_h = \sum_s s\,\sigma\left(s|h;\beta\right)$. The above derived equations rely on a matching between activity correlations, subject to different intensities of the external field.

The parameters of the network are then required to adapt in order to encode the progressively vanishing and driving effect of the external field. We should here notice that, the equation relative to thresholds is strictly necessary only in the sparse neuronal model, being the notion of threshold in the ± 1 model redundant, unless we want to introduce some kind of robustness.

C.1.1 Kullback-Leibler divergence and log-pseudo-likelihood

In the specific case of strongly biasing external signals suddenly vanishing to zero, namely $\lambda_1 \to \infty$ and $\lambda_2 = 0$, optimizing the KL divergence is equivalent to optimize an on-line version of the log-pseudo-likelihood. As we have already anticipated in the main text, the log-pseudo-likelihood can be viewed as an approximation of the log-likelihood, when the joint probability distribution of the input data is unknown and therefore approximated by means of a product of conditional probabilities over the neuronal states:

$$P\left(\boldsymbol{s} = \boldsymbol{\xi}^{\mu}\right) = \prod_i P\left(s_i = \xi_i^{\mu} | \left\{ s_j = \xi_j^{\mu} \right\}_{j \neq i}\right). \tag{C.6}$$

Replacing the expression of the probability distribution over all possible initial states in the case of strongly biasing signals, namely

$$P_{\text{clamp}}\left(\boldsymbol{s}; \boldsymbol{\xi}\right) = \prod_{i=1}^{N} \delta_{s_i, \xi_i}, \tag{C.7}$$

in the definition of the average KL divergence, we get:

$$\begin{aligned} &\left\langle \text{KL}\left[P\left(\cdot|\boldsymbol{s}; \lambda^{\text{ext}} = \infty\right) || P\left(\cdot|\boldsymbol{s}; \lambda^{\text{ext}} = 0\right)\right] \right\rangle_{P_{\text{clamp}}(\boldsymbol{s}; \boldsymbol{\xi})} = \\ &= -\sum_{i=1}^{N} \log P\left(s_i = \xi_i | \left\{ s_j = \xi_j \right\}_{j \neq i}; \lambda^{\text{ext}} = 0\right). \end{aligned} \tag{C.8}$$

Then, summing over all the patterns of activity that we want to store within the network, we finally obtain the exact expression of an on-line version of the log-pseudo-likelihood:

$$\mathscr{L}\left(\{\boldsymbol{\xi}^{\mu}\}|W_{ij},\theta;\beta\right)=\frac{1}{M}\sum_{\mu=1}^{M}\sum_{i=1}^{N}\log P\left(s_i=\xi_i^{\mu}|\left\{s_j=\xi_j^{\mu}\right\}_{j\neq i};\lambda^{ext}=0\right). \tag{C.9}$$

The optimization of the log-pseudo-likelihood can be equivalently viewed as a patterns stability criterion, since it enhances the probability for the network dynamics to remain confined in the basins of attraction of a given pattern $\boldsymbol{\xi}^{\mu}$, even in absence of the external field.

C.2 Inhibitory Schemes

In this section, we describe three different inhibitory schemes that we exploit in order to mimic the effect of the inhibitory network. Indeed, it is known that, when the synaptic couplings of the excitatory network are constrained to take a positive sign and when the neuronal states are described by sparse neuronal models, an inhibitory network is necessary for regulating the activity of the excitatory one, thus preventing the dynamics to collapse in the trivial states of all on or all off neurons.

C.2.1 A global inhibitory unit

This scheme relies on the introduction of a global inhibitory unit, that has to replace the effect of an entire network of inhibitory neurons. Here we propose the same scheme of Ref. [116], but extended to a more general setting.

To this end, we consider a number G of different excitatory neuronal species, each one constituted by a number N_α of neurons, such that the total number of excitatory neurons is given by $N=\sum_{\alpha=1}^{G}N_\alpha$. In order to keep the activity of each group of neurons, namely $S^\alpha=\sum_{i=1}^{N}s_i^\alpha$, to the desired level f_aN, we introduce G different global inhibitory units. The way through which the inhibitory units accomplish this task, is to send a feed-back signal, namely $\mathscr{I}^\alpha\left(\left\{f^\beta,S_\beta\right\}_{\beta=1}^{G}\right)$, to each group α of neurons. The feed-back signal elastically drives the activity of each sub-population of neurons towards the desired activity level, according to the current activity of both the neurons within the same sub-population, and the ones belonging to the other ones. We write the feed-back signal as:

$$\mathscr{I}^{\alpha}\left(\left\{f^{\beta},S_{\beta}\right\}_{\beta=1}^{G}\right)=H_{0}^{\alpha}+\nu^{\alpha\alpha}\left(S_{\alpha}-f^{\alpha}N_{\alpha}\right)+\sum_{\beta\neq\alpha}\nu^{\alpha\beta}\left(S_{\beta}-f^{\beta}N_{\beta}\right), \quad \text{(C.10)}$$

where H_0^{α} is a constant related to the basal inhibitory signal, namely when the activity of the sub-population of neurons already coincides with the desired one, while $\nu^{\alpha\beta}$ quantifies the intensity of the elastic signal. We need to determine these two quantities in order to fully identify the inhibitory contribution.

To this end, we suppose that the local fields h_i^{α} are Gaussian distributed. Then, if we want exactly $f^{\alpha}N_{\alpha}$ neurons to be active, we need to center the mean of the Gaussian distribution around the mean threshold $T^{\alpha}=\left\langle\theta_i^{\alpha}\right\rangle$, such that it holds:

$$\left\langle h_i^{\alpha}\right\rangle = T^{\alpha}-H^{-1}\left(f^{\alpha}\right)\sigma_{\alpha}, \quad \text{(C.11)}$$

where $H^{-1}(x)=\sqrt{2}\mathrm{erfc}^{-1}(2x)$ is the inverse error function, denoting how many standard deviation σ_{α} of the Gaussian distribution are needed in order to exactly find $f^{\alpha}N_{\alpha}$ local fields above threshold.

In the left hand side of Eq. (C.11), we can compute the mean value $\left\langle h_i^{\alpha}\right\rangle$ by relying on the definition of the local fields as a sum of three different contributions: the usual two contributions, namely the external field and the recurrent contribution from surrounding neurons, and the additional term constituted by the feed-back inhibitory signal:

$$\begin{aligned}\left\langle h_i^{\alpha}\right\rangle = \Bigg\langle & h_i^{\mathrm{ext},\alpha}+\sum_{j\neq i}W_{ij}^{\alpha\alpha}s_j^{\alpha}+\sum_{\beta\neq\alpha}\sum_{j}W_{ij}^{\alpha\beta}s_j^{\beta}-H_0^{\alpha}+ \\ & -\nu^{\alpha\alpha}\left(S^{\alpha}-f^{\alpha}N^{\alpha}\right)-\sum_{\beta\neq\alpha}\nu^{\alpha\beta}\left(S^{\beta}-f^{\beta}N_{\beta}\right)\Bigg\rangle.\end{aligned} \quad \text{(C.12)}$$

If we now sum and subtract the term $\sum_{\beta}\overline{W^{\alpha\beta}}S^{\beta}$ in the right hand side of Eq. (C.12), we get:

$$\begin{aligned}\langle h_i^\alpha \rangle = \overline{h^{\text{ext},\alpha}} + \overline{W^{\alpha\alpha}}\left(S^\alpha - f^\alpha\right) - \sum_{\beta\neq\alpha} \overline{W^{\alpha\beta}} S^\beta - H_0^\alpha + \\ - \nu^{\alpha\alpha}\left(S^\alpha - f^\alpha N^\alpha\right) - \sum_{\beta\neq\alpha} \nu^{\alpha\beta}\left(S^\beta - f^\beta N_\beta\right).\end{aligned} \tag{C.13}$$

Finally, we can compute the standard deviation σ_α in the right hand side of Eq. (C.11), by relying on both the definitions of local fields and variances of probability distributions in terms of squares of mean values. We thus obtain:

$$\sigma_\alpha = \sqrt{\left(\sigma_W^{\alpha\alpha}\right)^2\left(S^\alpha - f^\alpha\right) + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2 S^\beta}, \tag{C.14}$$

with $\sigma_W^{\alpha\beta}$ being the standard deviation associated to the synaptic couplings distribution. We can now expand σ_α around the desired activity level $f^\alpha N_\alpha$, thus getting:

$$\begin{aligned}\sigma_\alpha = \sqrt{f^\alpha\left(N_\alpha - 1\right)\left(\sigma_W^{\alpha\alpha}\right)^2 + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2 f^\beta N_\beta}\times \\ \times\left(1 + \frac{\left(\sigma_W^{\alpha\alpha}\right)^2\left(S^\alpha - f^\alpha N_\alpha\right) + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2\left(S^\beta - f^\beta N_\beta\right)}{2\left(f^\alpha\left(N_\alpha - 1\right)\left(\sigma_W^{\alpha\alpha}\right)^2 + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2 f^\beta N_\beta\right)}\right),\end{aligned} \tag{C.15}$$

where, in the last expression, we have further summed and subtracted the term $\sum_\beta\left(\sigma_W^{\alpha\beta}\right)^2 f^\beta N_\beta$ within the square root, before performing the expansion. Plugging Eq. (C.13) and Eq. (C.15) into Eq. (C.11), we finally get both H_0^α and $\nu^{\alpha\beta}$:

$$
\begin{aligned}
H_0^\alpha &= \overline{h^{\text{ext},\alpha}} + (N_\alpha - 1)\,\overline{W^{\alpha\alpha}} f^\alpha + \sum_{\beta\neq\alpha} N_\beta \overline{W^{\alpha\beta}} f^\beta + \\
&+ H^{-1}(f^\alpha)\sqrt{f^\alpha (N_\alpha - 1)\left(\sigma_W^{\alpha\alpha}\right)^2 + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2 f^\beta N_\beta} - T^\alpha \\
\nu^{\alpha\alpha} &= \overline{W^{\alpha\alpha}} + \frac{H^{-1}(f^\alpha)\left(\sigma_W^{\alpha\alpha}\right)^2}{2\sqrt{f^\alpha (N_\alpha - 1)\left(\sigma_W^{\alpha\alpha}\right)^2 + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2 f^\beta N_\beta}} \\
\nu^{\alpha\beta} &= \overline{W^{\alpha\beta}} + \frac{H^{-1}(f^\alpha)\left(\sigma_W^{\alpha\beta}\right)^2}{2\sqrt{f^\alpha (N_\alpha - 1)\left(\sigma_W^{\alpha\alpha}\right)^2 + \sum_{\beta\neq\alpha}\left(\sigma_W^{\alpha\beta}\right)^2 f^\beta N_\beta}}.
\end{aligned}
\tag{C.16}
$$

We need here to point out that, the modulation of the synaptic efficacy through the DCM learning rule is assumed to be an adiabatic process. Therefore, the mean of the synaptic weights, namely $\overline{W^{\alpha\beta}}$, and the variance, namely $\sigma_W^{\alpha\beta}$, are not required to adapt during the training phase, since their fluctuations take place at very long time scales.

Moreover, in the equation related to the basal inhibitory constant H_0^α, it is present a contribution coming from the external field, namely $\overline{h^{\text{ext},\alpha}}$. This simply implies that, those neurons that are not subject to any external field are kept fixed to their quiescent state.

We derived these equations in a quite general context. The fully-visible case can then be obtained by simply considering one single sub-population, namely $G = 1$, whereas the visible-hidden case can be recovered by considering two different sub-populations of neurons, namely the visible and the hidden neurons. In this case, $G = 2$.

C.2.2 The soft winner-takes-all scheme

We design the soft winner-takes-all scheme directly acting on the network dynamics, described by the Glauber transition probability in Eq. (4.1). Before the dynamics drives the network towards the next configuration of neuronal activity, all the local fields are sorted in an increasing order, according to their magnitudes. An inhibitory

signal then sets the spiking threshold just below the fN-th local field. This scheme presents the advantage of having more control on the precise value of the network current activity.

In the case of visible-hidden units, we separately apply the soft winner-takes-all scheme to both populations of neurons.

C.2.3 The spiking threshold modulation scheme

The adaptive threshold inhibitory scheme arises from the exact mapping between the ± 1 and sparse neuronal states model, namely $\hat{s}_i \to s_i = 2\hat{s}_i - 1$, with $\hat{s}_i \in \{0,1\}$ and $s_i \in \{-1,1\}$. In the sparse neuronal states model, the local field takes into account the further contribution of the local threshold:

$$\hat{h}_i = \hat{h}_i^{\text{ext}} + \sum_j \hat{W}_{ij}\hat{s}_i - \hat{\theta}_i. \tag{C.17}$$

According to the mapping mentioned above, provided that it holds $\hat{h}_i^{\text{ext}} = 2h_i^{\text{ext}}$, $\hat{W}_{ij} = 4W_{ij}$ and $\hat{\theta}_i = 2\sum_j W_{ij} = \frac{1}{2}\sum_j \hat{W}_{ij}$, the local field of the sparse neuronal states model in Eq. (C.17), can be written in terms of the parameters of the ± 1 neuronal states model as:

$$\hat{h}_i = 2\left(h_i^{\text{ext}} + \sum_j W_{ij}s_i\right). \tag{C.18}$$

Notice that the condition on the local threshold, namely

$$\hat{\theta}_i = \frac{1}{2}\sum_j \hat{W}_{ij}, \tag{C.19}$$

in the case we set the desired mean activity of the network to $f = 1/2$, translates into:

$$\hat{\theta}_i = f\sum_j \hat{W}_{ij} = \left\langle \sum_j \hat{W}_{ij}\hat{s}_i \right\rangle. \tag{C.20}$$

Because of that, we can then rewrite the local fields of the sparse neuronal states model as:

$$\begin{aligned} \hat{h}_i &= \hat{h}_i^{\text{ext}} + \sum_j \hat{W}_{ij}\hat{s}_i - \hat{\theta}_i \\ &= \hat{h}_i^{\text{ext}} + \sum_j \hat{W}_{ij}\left(\hat{s}_i - f\right). \end{aligned} \tag{C.21}$$

This induces a slightly change in the definition of the DCM learning rule, namely

$$\Delta W_{ij} \propto \left(\left\langle s_i^{t+1}\left(s_j^t - f\right)\right\rangle_{t,\lambda_1} - \left\langle s_i^{t+1}\left(s_j^t - f\right)\right\rangle_{t,\lambda_2} \right). \tag{C.22}$$

Therefore, for the exact mapping to hold, the thresholds are required to adapt in light of the modulation of the synaptic couplings. Even if this holds only when $f = 1/2$, it reveals to be necessary in the on-line learning regime in order to achieve an extensive palimpsest capacity.

C.3 TAP Approach

The DCM learning rule relies on the matching between activity correlations in presence of different intensities of the external field. Estimating correlations in kinetic models may be not an easy task. Indeed the lack of an Hamiltonian, does not allow to straightforwardly take advantage of the usual Mean Field methods. As pointed out in the main text, in the more biologically plausible setting, we directly estimate these correlations from the network dynamics, through Monte Carlo Markov Chains (MCMCs) and assuming ergodicity to hold. However, in kinetic models, correlations can be also computed by means of a mean-field method, that goes under the name of Thouless-Anderson-Palmer (TAP) approach. This has been shown in Ref. [110].

The advantage of exploiting a Monte Carlo sampling instead of resorting to the TAP approach is that, through MCMCs, we are actually miming the network dynamics. Conversely, through the TAP approach, we can provide better estimates, relying on the computation of the marginals of the neuronal states variables. Given the marginals, we can then easily determine both averages and pairwise correlations.

The method proposed in Ref. [110] has been derived in the context of $s_i \in \{-1,1\}$ spin model and asynchronous Glauber dynamics. Here we present our extension to sparse neuronal states models, namely $s_i \in \{0,1\}$, and synchronous Glauber dynamics.

As already pointed out in the main text, when dealing with kinetic models, the explicit expression of the steady-state distribution, assuming that a stationary state exists, is unknown. However, in the manifold of all possible probability distributions, we can choose to approximate the joint distribution of the stationary state, namely $P(s|h,W)$, with the single-site factorized distribution:

$$P^{\mathrm{MF}}\left(s|h^{\mathrm{MF}}\right) = \prod_{a=1}^{N} \frac{\exp\left(h_a^{\mathrm{MF}} s_a\right)}{1+\exp\left(h_a^{\mathrm{MF}}\right)}. \tag{C.23}$$

The above Mean Field approximation holds if assuming a weakly interacting regime, where the synaptic couplings are of order $1/\sqrt{N}$. If $P(s|h,W)$ is the unknown stationary distribution, then we can think to expand it around the Mean Field limit, namely $h^{\mathrm{MF}} = h - dh$ and $W^{\mathrm{MF}} = dW$, whit h and W being the parameters of the stationary probability distribution. Notice that, in our model, W are the synaptic couplings and h corresponds to the local fields, given by the sum of two contributions: the external field and the thresholds, namely $h_a \to \lambda^{\mathrm{ext}}\left(\xi_a - \frac{1}{2}\right) - \theta_a$.

We then want to determine the configuration of h and W, optimizing the KL divergence between $P(s|h,W)$ and its Mean Field approximation, namely $P^{\mathrm{MF}}\left(s|h^{\mathrm{MF}}\right)$:

$$KL\left[P||P^{\mathrm{MF}}\right] = \sum_{\{s\}} P(s|h,W) \log\left(\frac{P(s|h,W)}{P^{\mathrm{MF}}\left(s|h^{\mathrm{MF}}\right)}\right). \tag{C.24}$$

The optimization of the KL divergence in Eq. (C.24) leads to a matching condition for the first moments of the two distributions, namely

$$m_a - m_a^{\mathrm{MF}} = 0 \qquad \forall a = 1,...,N, \tag{C.25}$$

whit $m_a = \sum_{\{s\}} P(s_a)\, s_a$. Expanding the first moment of $P(s|h,W)$ up to the second order, we get:

$$
\begin{aligned}
0 = m_a - m_a^{\mathrm{MF}} \approx & \sum_i \left.\frac{\partial m_a}{\partial h_i}\right|_{\mathrm{MF}} dh_i + \sum_{i<j} \left.\frac{\partial m_a}{\partial W_{ij}}\right|_{\mathrm{MF}} dW_{ij} + \\
& + \sum_{ij} \left.\frac{\partial^2 m_a}{\partial h_i \partial h_j}\right|_{\mathrm{MF}} dh_i dh_j + \sum_{i<j}\sum_{k<l} \left.\frac{\partial^2 m_a}{\partial W_{ij} \partial W_{kl}}\right|_{\mathrm{MF}} dW_{ij} dW_{kl} + \\
& + 2 \sum_{i<j}\sum_{k} \left.\frac{\partial^2 m_a}{\partial W_{ij} \partial h_k}\right|_{\mathrm{MF}} dW_{ij} dh_k,
\end{aligned}
$$

where the derivatives can be explicitly computed, thus leading to the following set of equations:

$$
\left.\frac{\partial m_a}{\partial h_i}\right|_{\mathrm{MF}} = m_a (1 - m_a)\, \delta_{ai}
$$

$$
\left.\frac{\partial m_a}{\partial W_{ij}}\right|_{\mathrm{MF}} = m_j m_a (1 - m_a)\, \delta_{ai}
$$

$$
\left.\frac{\partial^2 m_a}{\partial h_i \partial h_j}\right|_{\mathrm{MF}} = \Big(m_a (1 - m_a)^2 - (m_a)^2 (1 - m_a) \Big)\, \delta_{ai} \delta_{aj}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 m_a}{\partial W_{ij} \partial h_k}\right|_{\mathrm{MF}} = & \, m_j m_a \left(1 - m_j\right) (1 - m_a)\, \delta_{ai} \delta_{jk} + \\
& + m_j \Big[m_a (1 - m_a)^2 - (m_a)^2 (1 - m_a) \Big]\, \delta_{ai} \delta_{ak}
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 m_a}{\partial W_{ij} \partial W_{kl}}\right|_{\mathrm{MF}} = & \, m_j m_l (1 - m_l)\, m_a (1 - m_a)\, \delta_{li} \delta_{ak} + \\
& + m_l m_j \left(1 - m_j\right) m_a (1 - m_a)\, \delta_{jk} \delta_{ai} + \\
& + \left\langle s_j s_l \right\rangle_{\mathrm{MF}} \Big(m_a (1 - m_a)^2 - (m_a)^2 (1 - m_a) \Big)\, \delta_{ai} \delta_{ak}.
\end{aligned}
$$

At this point, if we further exploit the identity $\left\langle s_j s_l \right\rangle_{\mathrm{MF}} = \delta_{jl} m_j + \left(1 - \delta_{jl}\right) m_j m_l$ and we disregard all those contributions in the expansion higher than the second order, the matching condition finally reads:

$$h_a^{\mathrm{MF}} = h_a + \sum_j m_j W_{aj} + \frac{1}{2}(1-2m_a)\sum_j \left(m_j\left(1-m_j\right)\right) W_{aj}^2. \tag{C.26}$$

The resulting first moments can then be derived by applying the sigmoid function, namely $\mathrm{sigm}(x) = (1+e^{-x})^{-1}$, to the above equation, thus obtaining the exact expression of the TAP equations:

$$m_i = \mathrm{sigm}\left(\theta_i + \sum_j m_j W_{ij} - \left(m_i - \frac{1}{2}\right)\sum_j \left(m_j\left(1-m_j\right)\right) W_{ij}^2\right). \tag{C.27}$$

This is a self-consistency equation that needs to be solved recursively, starting from an initial value of the magnetizations and then iterating up to a fixed point is reached. The connected correlations are then given by:

$$\chi_{ij}^D = \left\langle s_i' s_j \right\rangle - m_i m_j = \left\langle s_i \left(\mathrm{sigm}\left(h_j\right) - m_j\right)\right\rangle, \tag{C.28}$$

where the first moments $m_i \ \forall i = 1,...,N$ are provided by Eq. (C.27), whereas the disconnected correlations are computed as usual as:

$$\begin{aligned} \left\langle s_i' s_j \right\rangle &= \sum_{\{s\}} P(s)\, s_j \sum_{s_i'} P\left(s_i'|s\right) s_i' \\ &= \left\langle \mathrm{sigm}\left(h_i\right) s_j \right\rangle, \end{aligned} \tag{C.29}$$

where in the last equation, we explicitly sum over all the possible neuronal states. As it is for the magnetizations, even this quantity can be estimated by means of a second order expansion in the parameters of the stationary distribution, thus leading to the following set of equations:

$$\left.\frac{\partial \chi^D_{ba}}{\partial h_i}\right|_{\text{MF}} = 0$$

$$\left.\frac{\partial \chi^D_{ba}}{\partial W_{ij}}\right|_{\text{MF}} = m_b m_a \left(1-m_b\right)\left(1-m_j\right)\delta_{aj}\delta_{bi}$$

$$\left.\frac{\partial^2 \chi^D_{ba}}{\partial h_i \partial h_j}\right|_{\text{MF}} = 0 \tag{C.30}$$

$$\left.\frac{\partial^2 \chi^D_{ba}}{\partial W_{ij} \partial h_k}\right|_{\text{MF}} = m_a m_b \left(1-m_a\right)\left(1-m_b\right)\left(1-2m_b\right)\delta_{aj}\delta_{bk}\delta_{bi}$$

$$\begin{aligned}\left.\frac{\partial^2 \chi^D_{ba}}{\partial W_{ij} \partial W_{kl}}\right|_{\text{MF}} &= m_a m_b \left(1-m_b\right)\left(1-2m_b\right)\delta_{bk}\delta_{bi}\times \\ &\times\left(\delta_{aj}\delta_{al}+\left(1-\delta_{aj}\right)\delta_{al}m_j\left(1-m_a\right)\right).\end{aligned}$$

Then, taking advantage of the identity:

$$\begin{aligned}\left\langle s_a s_j s_l\right\rangle_{\text{MF}} &= \delta_{aj}\left(\delta_{al}m_a+\left(1-\delta_{al}\right)m_a m_l\right)+ \\ &+\left(1-\delta_{aj}\right)\left(\delta_{al}m_a m_j+\left(1-\delta_{al}\right)m_a\left\langle s_j s_l\right\rangle_{\text{MF}}\right),\end{aligned} \tag{C.31}$$

we can finally obtain the equations for the connected correlations in light of the magnetizations:

$$\chi^D_{ij} = \left(m_i\left(1-m_i\right)\right)\left(m_j\left(1-m_j\right)\right)\left(J_{ij}+\frac{1}{2}\left(2m_i-1\right)\left(2m_j-1\right)\left(W_{ij}\right)^2\right), \tag{C.32}$$

The disconnected correlations can then be straightforwardly determined relying on the connected ones:

$$\begin{aligned}\left\langle s_i' s_j \right\rangle = &\left(m_i\left(1-m_i\right)\right)\left(m_j\left(1-m_j\right)\right)\times \\ &\times\left(W_{ij}+\frac{1}{2}\left(2m_i-1\right)\left(2m_j-1\right)\left(W_{ij}\right)^2\right)+m_i m_j.\end{aligned} \tag{C.33}$$

C.4 Simulations Details

C.4.1 The training phase

In this section we provide the details concerning both the implementation and simulations of the training process, during which the network learns how to store a given number of patterns, on the base of the DCM learning rule. The learning protocol consists in adapting both the synaptic couplings and thresholds, according to Eq. (4.7). At the beginning of the training phase, in case of ± 1 neuronal states, we uniformly initialize the synaptic couplings within the range $\left[-\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}\right]$ and we set the thresholds all to zero. Conversely, in the case of sparse neuronal states, we initialize the synaptic couplings within the range $\left[0, \frac{1}{\sqrt{N}}\right]$ and we set all the thresholds to the constant value 0.3. We need here to point out that, because of the role played by the inhibitory mechanisms, the initialization of the thresholds in the latter neuronal model is actually redundant.

During the training, for every pattern $\boldsymbol{\xi}^\mu$, we initialize the intensity of the external field at the starting value $\lambda = \lambda^{max}$ and we perform T_{init} steps of Glauber dynamics. As said, this is needed for centering the activity of the network around the pattern $\boldsymbol{\xi}^\mu$. Then, we start recording the time-delayed correlations for a certain number T of steps, under the presence of an external field of intensity λ. After that, we decrease the external field of $\Delta\lambda$ and we record the same correlations for another number T of time steps. We finally update the synaptic couplings and the thresholds according to the update rules of Eq. (4.7). We repeat this schedule for every pattern and for a certain number of epochs, that we typically set to 1000, even if, for finite external fields we lowered it up to 250.

The number of steps needed for recording the time-delayed correlations, namely T, has to be chosen in such a way to ensure that, at the end of the T steps, the activity of the network is still close to the pattern $\boldsymbol{\xi}^\mu$, otherwise we would need to re-initialize the state of the network. In our simulations, we always set T within the

range $[3,25]$, while we kept the learning rate η constant, typically at 0.01. The fact that the DCM learning rule is capable of good performances, even when choosing very small waiting times T, clearly points out that it is able to work even in very noisy contexts. However, as long as we consider smaller waiting times, the learning rate needs to be reduced as well. Moreover, because the updates of the network parameters are purely local, they can be entirely performed in parallel.

We provide a pseudo-code of the above described learning schedule during the training phase in Algo. (C.1).

Input: parameters: η, *cycles*, λ_{max}, $\Delta\lambda$, T, T_{init}
Initialize W randomly $\sim U\left(\frac{-1}{\sqrt{N}}, \frac{1}{\sqrt{N}}\right)$;
for *cycle* $= 1$ **to** *cycles* **do**
  for μ *in random permutation of* $[1:p]$ **do**
    Set the external field on the visible neurons to an intensity λ_{max};
    Run the network for T_{init} steps;
    while $\lambda > 0$ **do**
      Estimate $\left\langle s_i^{t+1} s_j^t \right\rangle_\lambda$ for T steps;
      Estimate $\left\langle s_i^{t+1} s_j^t \right\rangle_{\lambda-\Delta\lambda}$ for T steps;
      $W_{ij} \leftarrow W_{ij} + \eta \left[\left\langle s_i^{t+1} s_j^t \right\rangle_\lambda - \left\langle s_i^{t+1} s_j^t \right\rangle_{\lambda-\Delta\lambda} \right]$;
      $\lambda \leftarrow \lambda - \Delta\lambda$;
    end
  end
  if *all patterns are learned* **then**
    BREAK;
  end
end

Fig. C.1 Pseudo-code of the learning protocol described in the main text. The synaptic couplings are updated according to the DCM learning rule. In the pseudo-code, $s_i \in \{-1,1\}$.

C.4.2 The retrieval phase

In order to check if a pattern $\boldsymbol{\xi}^\mu$ has been effectively stored by the network during the training phase, we initialize the state of the network to a noisy version of $\boldsymbol{\xi}^\mu$. Denoting with χ the level of noise according to which the original pattern $\boldsymbol{\xi}^\mu$ has been corrupted, we can say that the pattern has been successfully stored by the network if, initializing the state of the network close to $\boldsymbol{\xi}^\mu$ with a noisy level of χ, and then let it to evolve for 50 steps of Glauber dynamics, the evolution of the

network ends up in the attractor state $\boldsymbol{\xi}^{\mu}$ the 90% of times, over 100 restarts of the network dynamics.

To verify if the network dynamics has actually reached a fixed point, after 50 steps of Glauber dynamics, we measure the overlap between the pattern $\boldsymbol{\xi}^{\mu}$ and the final configuration of the network. If this overlap is greater than 0.99, then we say that $\boldsymbol{\xi}^{\mu}$ has been actually stored by the network.

C.4.3 Detection of spurious attractors

As already pointed out in the main text, one of the main drawbacks of the Hebbian learning is its tendency of generating spurious attractors. In order to detect them, we exploit the following strategy. First of all, we chose a value of the storage load α far from the critical capacity of both Hebbian learning and the DCM learning rule, thus allowing both of them to properly work. Then, we randomly initialize the state of the network. After 200 steps of Glauber dynamics, we check if the reached state of the network shows an overlap ≥ 0.95 with one of the patterns that we want to store. If this is not the case, we simulate the dynamics for another number of time steps and then we check if it has reached a stationary state. If this happens, then we add one spurious attractor to the counter.

This method provides only a numerical estimate of the number of spurious attractors, however, it reveals to be already sufficient for pointing out the discrepancy between the DCM learning rule and the Hebbian learning rule, as shown in Fig. (4.7).

C.4.4 On-line learning

In the on-line learning regime, instead of repeatedly showing a set of patterns for a given number of epochs, we present to the network only one pattern at a time, until it is successfully stored by the network. Then we switch to the next pattern. This procedure goes on until we reach a stationary condition, where the number of stored patterns stays constant, meaning that the most recently showed pattern has actually replaced one of the previously stored ones. This condition is reached at the palimpsest capacity.

To reach an extensive palimpsest capacity, we need to closely tune all the parameters of the simulation, such that the learning process slows down and the presentation of the new patterns does not consistently affect the already stored ones. In particular, we need to be careful especially in the last time window, where there is no more the driving signal from the external field. In this phase, the dynamics of the network can actually move very far away from the basin of attraction of the pattern that we are trying to store. When this is the case, it can further converge towards the basins of the already stored ones, thus causing, most of the time, their loss. We tried to remove the last time window from the training phase, but this only weakly improved the palimpsest capacity.

Moreover, in the sparse neuronal model, namely $s_i \in \{0,1\}$, we are not able to reach an extensive palimpsest capacity, unless we modify the DCM learning rule, by taking into account the adaptive thresholds inhibitory scheme (see Eq. (Eq. (C.22))). This may be thus related to the choice of the thresholds. Indeed, it seems that we need to shift the thresholds such that we can get wider basins of attraction. This problem does not arise in the non on-line learning regime, because cycling over all the patterns many times, makes the threshold to set on values that are consistent with all the patterns.

We choose the following values for the parameters in the on-line regime: $\eta = 0.01$, $\lambda_{max} = 4$, $\Delta\lambda = 1$ and $T = 10$. According to this parameters, the number of times needed to the network for storing a new pattern is around 1000. Due to the increase of the average strength of the synaptic couplings, as the learning phase goes on, this number gets larger and larger as new patterns stabilize. We can overcome this problem by introducing a regularization effect on the synaptic couplings.

C.4.5 RBMs on MNIST

In this section, we provide the detail concerning the simulations we performed for testing the performance of the DCM learning rule, when a set of hidden neuronal states is added to the network. In particular, we consider the specific architecture of RBMs. In our simulation, we choose as input patterns the ones in the MNIST data set. They represent a set of 7×10^4 grayscale images of handwritten digits, encoded in an array of 28×28 pixels and grouped in 10 different classes, each one representing a number from 0 to 9. We exploited 6×10^4 of these images for training

the network and 1×10^4 for testing its generalization performances on classification tasks.

The MNIST images are characterized by an average number of turned on pixels of $\bar{f}_\xi = 0.13066$, where each pixel ξ_i^μ takes values within $[0,1]$. To test the DCM plasticity rule on different learning tasks, we have thus constructed a stochastic neural network made of $|V| = 784 + 10$ visible units, and $|H| = 1000$ hidden units, in order to enhance the representational power of the network. We add an extra layer of 10 units, for the implementation of classification tasks.

We applied our new learning protocol by cycling 2 times over the whole training set, basically performing two different experiments:

- *strong external fields*. We tested the performances of the DCM rule in presence of very strong biasing signals, such that the learning problem can be considered as equivalent to the optimization problem of an on-line version of the log-pseudo-likelihood. In this case we set $\lambda^{max} = 50$ and then we suddenly drop the intensity of the external field to zero;

- *weak external fields*. In this experiment, we applied the DCM learning rule on a more biological setting, where the initial strength of the external field is set to $\lambda^{max} = 3$ and then gradually decreased to zero of an amount of $\Delta\lambda = 3/2$. We fix the desired activity level of the visible neurons to the average number of turned on pixels in the MNIST images, while we set the one of the hidden neurons at $f_h = 0.2$. We choose the soft winner-takes-all mechanism as inhibitory scheme. Given that, we simulate the training phase, with a number $T = 15$ steps of Glauber dynamics. Notice that, the level of the noise, affecting the dynamics and codified by β, needs to be properly chosen, since thanks to that we can break the possible arising symmetries among hidden units.

Features Extraction

As pointed out in the main text, hidden neuronal states are usually employed in order to increase the representational power of neuronal networks. Indeed, they are left free to catch not trivial correlations in the structure of the training data. Hidden units are thus often identified as features detectors of the input patterns. Indeed, each

hidden unit experiences a field exerted by all the surrounding neurons. This field, usually known as the *receptive field*, in the specific case of RBMs, is given by:

$$h_i = \sum_{j \neq i} W_{ij} s_j^v, \tag{C.34}$$

with $i \in H$ and $j \in V$. During the training phase, the receptive fields specialized in encoding specific features of the pattern, so that they set above threshold whenever those features are observed in the input pattern, thus exciting the corresponding hidden unit. In this way the RBM is able to construct an internal representation of the incoming stimulus, by encoding it in the synaptic couplings.

The receptive fields can be organized into a matrix of dimensions 28×28 grayscale pixels and then normalized to binary pixels $\{0, 1\}$.

In the main text, we have shown some of the receptive fields or features, extracted by an RBM when trained according to the DCM learning rule. Among them, we can find very noisy features, that however, do not affect the performances of the network, being them unable to exceed the value set by the activation threshold.

Generation of new samples: Gibbs Sampling

Once an RBM has been trained to build an internal representation of the input data, it can be exploited to generate new data samples, distributed according to the same statistics of the input data. We have thus tested the DCM learning rule on this further task in the following way. In the training phase, we first clamp the initial configuration of the visible neurons into one of the input patterns, by applying an external field of intensity $\lambda = 50$ and letting the dynamics of the network to evolve, under the presence of such a strong biasing signal. After 30 steps of Glauber dynamics, we remove the external field on the 784 visible neurons, while still keeping a supervised signal on the 10 output units. This signal induces the network to produce samples that are distributed according to the same statistics of the corresponding input pattern. Then, we let the dynamics to evolve, according to Gibbs chains, and we look if the stationary distribution reached by the network is close to the statistics of the input pattern. When this is the case, a new sample of the input data is generated.

Classification task

In order to test the classification performance of an RBM when trained through the DCM learning rule, we present to the network the input stimulus, by clamping the initial configuration of the visible neurons to a given input pattern. We then look if the resulting distribution of the output units is peaked on the label corresponding to the input. As usual, we have tested the performance of the learning rule on two different experiments: the first one where the time-delayed correlations are computed through the TAP approach, thus looking at the output unit with the highest magnetization, and the other one where they are instead directly estimated from the network dynamics.