Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Dipartimento di Fisica e Astronomia
Corso di Laurea Magistrale in Fisica

Applicazione di un algoritmo d'apprendimento basato su sistemi fuori dall'equilibrio a dati di Genome Wide Association

Relatore:
Prof. Gastone Castellani

Correlatore:
Dott. Nico Curti

Presentata da:
Daniele Dall'Olio

Anno Accademico 2018/2019

Prefazione

Il fenomeno dell'apprendimento può essere studiato attraverso metodiche di Meccanica Statistica unite alla cosiddetta Large Deviation Theory. In generale, l'apprendimento può essere suddiviso in due problemi: il problema di classificazione ed il problema di generalizzazione. Il primo mira a memorizzare completamente un insieme di associazioni input-output estratte casualmente (*apprendimento perfetto*). Il secondo, invece, equivale ad apprendere una regola che combini le proprietà di un oggetto in modo da renderlo identificabile. Sfruttando le Neural Networks come modello idealizzato del funzionamento dei neuroni, è possibile identificare entrambi i problemi come un sistema di spin interagenti e definire una forma di energia per tale sistema. A partire da una descrizione all'equilibrio del sistema (basata sulla distribuzione di Boltzmann) e sotto opportune condizioni, il problema di classificazione si dimostra computazionalmente complesso (*NP-completo*). Nonostante tale complessità, sono stati trovati una gamma di algoritmi euristici in grado di risolvere lo stesso problema efficacemente. É possibile dimostrare che questa apparente inconsistenza è dovuta al fatto che lo spazio delle soluzioni degli algoritmi euristici efficaci non coincida con quello tipico, atteso dalla descrizione all'equilibrio. In altre parole, le soluzioni degli algoritmi efficaci hanno proprietà differenti da quelle tipiche e risultano invisibili alla statistica all'equilibrio. Questo risultato apre all'utilizzo di distribuzioni fuori dall'equilibrio e all'integrazione di studi di Large Deviation Analysis (LDA). Tramite l'utilizzo di distribuzioni in grado di evidenziare le proprietà delle soluzioni fuori dall'equilibrio è possibile realizzare l'algoritmo *replicated focusing Belief Propagation* (rfBP), i cui risultati in termini di performance computazionali e di natura delle soluzioni sono in linea con i risultati degli algoritmi euristici efficaci. Tale affinità giustifica l'uso di effetti fuori dall'equilibrio e di metodiche di LDA nello studio del fenomeno dell'apprendimento.

Questo lavoro contestualizza l'apprendimento nell'ambito fisico-statistico e esplicita le connessioni tra i contenuti multidisciplinari coinvolti. Nella trattazione si evidenzia come l'utilizzo integrato di modelli a Spin-Glass, grafi e Neural Networks siano in grado di creare una base teorica solida per lo sviluppo di algoritmi di machine learning originali e innovativi. Il lavoro si è sviluppato con l'obiettivo di ricondursi dallo studio del fenomeno dell'apprendimento all'algoritmo rfBP e di applicarlo in un contesto reale-pratico.

Questo lavoro, inoltre, introduce una nuova libreria di C++ ottimizzata per il calcolo parallelo dell'algoritmo rfBP e applica tale algoritmo su dati di Genome Wide Association. In particolare, sono stati considerati campioni di genomi del batterio Salmonella, ospitati in diversi animali, ed è stato effettutato il training dell'algoritmo rfBP sull'insorgenza di mutazioni (*Single Nucleotide Polymorphism*, SNP), nel tentativo di determinare l'animale da cui essi sono stati ospitati. L'obiettivo di questa applicazione è capire come i genomi dei batteri siano influenzati dal proprio ospite animale e se è possibile evidenziare delle caratteristiche che permettano di risalire dalla sequenza di SNPs all'ospite. Riuscire a trovare queste caratteristiche permetterebbe di migliorare il controllo degli

animali che apportano delle modifiche potenzialmente infettanti al genoma del batterio. Una conoscenza di questo tipo favorirebbe di conseguenza il controllo delle infezioni ed il commercio di cibi più salutari. Questa applicazione è parte integrante del progetto COMPARE finanziato dall'Unione Europea all'interno del programma di ricerca e innovazione Horizon 2020 (*grant agreement* No. 643476).

Su questi dati, l'algoritmo rfBP produce ottimi risultanti sia in termini di *accuracy* che di *coefficiente di correlazione di Matthews* (MCC). Inoltre, questi risultati si dimostrano comparabili e superiori a quelli ricavati con le più comuni tecniche di Machine Learning.

Il lavoro è organizzato come di seguito. Nel primo capitolo sono esposte alcune nozioni di base di Meccanica Statistica, Neural Network Theory, Network Theory e Teoria della complessità computazionale. Nel secondo capitolo è descritto l'algoritmo di *Belief Propagation Standard*. Nel terzo capitolo sono riportati i caratteri generali dei sistemi di Ising e di Spin Glasses, con lo scopo di mostrare che esiste una corrispondenza tra le *magnetizzazioni cave* degli spin di tali sistemi e un'opportuna parametrizzazione delle equazioni di Belief Propagation. Nel quarto capitolo è contestualizzato lo studio del fenomeno dell'apprendimento nel campo fisico-statistico, passando da un'iniziale trattazione all'equilibrio ad una seguente trattazione fuori dall'equilibrio. Nel quinto capitolo è infine presentata la replicated focusing Belief Propagation.

In conclusione nel sesto capitolo sono mostrati i risultati ottenuti con l'algoritmo rfBP sui dati di genomica e il confronto di questo algoritmo rispetto alle tecniche più comuni di Machine Learning.

Indice

Capitolo 1

Introduzione

Questo capitolo introduce i tratti essenziali di Meccanica Statistica, Sistemi Complessi, Teoria dei Network e Complessità Computazionale fondamentali per comprendere le tematiche esposte nei prossimi capitoli.

Nella prima sezione sono introdotte alcune nozioni di Meccanica Statistica, dai sistemi termodinamici alla definizione di entropia.

La seconda sezione riassume i concetti introduttivi della Neural Network Theory con particolare attenzione alle Neural Networks feed-forward fully-connected.

La terza sezione introduce la Network Theory, focalizzandosi in particolare sui grafi bipartiti.

Nella quarta sezione infine sono esposti gli elementi fondamentali della Teoria della Complessità Computazionale.

1.1 Introduzione alla Meccanica Statistica

La Meccanica Statistica è una branca della fisica teorica che mira a descrivere le proprietà macroscopiche dei sistemi studiati in funzione della dinamica microscopica degli elementi che lo costituiscono [1–3].

I primi sviluppi di Meccanica Statistica sono stati compiuti sui sistemi termodinamici con l'intento di ricondursi dalle quantità microscopiche alle proprietà macroscopiche.

1.1.1 I Sistemi Termodinamici

Un sistema termodinamico è una porzione di volume nello spazio il cui contenuto in termini di materia ed energia è descrivibile attraverso le grandezze che caratterizzano la termodinamica. Questi sistemi sono costituiti da particelle che in ogni istante temporale si ritrovano in precise coordinate spaziali ($\vec{q}$) e sono dotate di un determinato *momento*

($\vec{p}$=$m\,\vec{v}$). Le informazioni sul momento e sulla posizione spaziale definiscono lo stato dinamico della particella.

I sistemi termodinamici sono trattati a partire da uno spazio in cui ogni punto rappresenta uno stato dinamico del sistema. Questo spazio equivale, in altre parole, all'insieme degli stati dinamici delle singole particelle ad un preciso istante temporale. Tali stati sono chiamati *microstati* e lo spazio a cui appartengono è detto *spazio delle fasi*. I microstati seguono la dinamica Hamiltoniana [1] e ad ognuno di essi corrispondono determinate proprietà macroscopiche osservabili, che determinano il cosiddetto *macrostato*.

La relazione tra microstato e macrostato non è univoca, dato che innumerevoli stati dinamici sono in relazione con le stesse proprietà macroscopiche. Perciò, un macrostato è riferito ad una collezione di sistemi che corrispondono a diversi microstati. Questa collezione di copie del sistema è detta *ensemble*. L'ensemble è descrivibile matematicamente attraverso una distribuzione di probabilità, che indica come il sistema si distribuisce sui possibili microstati.

1.1.2 Ensemble canonico

Tipicamente si assume che i sistemi termodinamici contengano in un volume V di grande dimensione un numero N elevato di particelle e, convenzionalmente, si studiano tali sistemi considerando il caso in cui, avendo $N \to \infty$ e $V \to \infty$, $\frac{N}{V} = costante$. Tale condizione è chiamata *limite termodinamico*.

Ora, si prendano in considerazione i sistemi termodinamici *isolati*, ovvero quei sistemi termodinamici che non scambiano energia e materia con l'ambiente. L'energia interna di tali sistemi si conserva o, più realisticamente, rimane compresa in un intervallo infinitesimo $[E, E + \delta E]$.

Nel tentativo di descrivere tali sistemi all'equilibrio termico [2] , la meccanica statistica si affida al postulato di ugual probabilità a priori, che prefissa l'equiprobabilità di tutti i microstati nell'ensemble compatibili con le proprietà macroscopiche emergenti. Tali microstati compatibili sono detti microstati *accessibili*. Nello specifico, l'ensemble, che è stazionario visto che il sistema è all'equilibrio, è chiamato *ensemble microcanonico* ed il macrostato dell'ensemble è definito dai valori fissati di E, V ed N.

Formalmente, l'ensemble microcanonico può essere quindi descritto da una distribuzione di probabilità che è costante e diversa da zero solo per i microstati accessibili. Da

[1] Equazioni di Hamilton:

$$\begin{cases} \dfrac{d\vec{q}}{dt} = \dfrac{\partial \mathcal{H}(t, \vec{q}, \vec{p})}{\partial \vec{p}} \\ \dfrac{d\vec{p}}{dt} = -\dfrac{\partial \mathcal{H}(t, \vec{q}, \vec{p})}{\partial \vec{q}} \end{cases} \tag{1.1}$$

in cui $\mathcal{H}(t, \vec{q}, \vec{p})$ è l'Hamiltoniana del sistema.

[2] Genericamente parlando, l'equilibrio termico indica lo stato di un sistema in cui sono assenti scambi di energia termica.

tale distribuzione è possibile calcolare il numero di questi microstati, definire l'entropia del sistema e derivare le quantità classiche di termodinamica.

Lo studio dei sistemi all'equilibrio termico si estende anche ai sistemi non isolati, per i quali non resta invariata l'energia interna ma la temperatura T. In questo caso la collezione di copie del sistema è definito *ensemble canonico* e il macrostato è definito da N, V e T. Siccome l'energia dei sistemi nell'ensemble non è fissata, questo studio affronta il problema di determinare ad un certo istante di tempo la probabilità che un sistema dell'ensemble abbia una certa energia.

Un approccio comune al problema consiste nell'analizzare un macrosistema isolato all'equilibrio, costituito da M copie del sistema, che ha energia totale E_{tot}. Tale energia può essere espressa come $E_{tot} = \sum n_i E_i$ così da sottolineare l'esistenza di n_i copie del sistema aventi energia E_i. Siccome il numero di copie nel macrosistema non varia, M si conserva e le variabili n_i devono rispettare la condizione $M = \sum n_i$. Il macrosistema può essere quindi rappresentato da qualsiasi configurazione $\{n_i\}$ che rispetti le condizioni precedenti per E_{tot} ed M.

Considerando una configurazione $\{n_i\}$, si ha un numero $W_{\{n_i\}}$ calcolabile statisticamente di possibili modi attraverso cui tale configurazione può essere realizzata dagli M sistemi. Dato che E_{tot} è conservata, il macrosistema può essere anche interpretato come un ensemble microcanonico e quindi ogni possibile configurazione $\{n_i\}$ è equamente probabile che si realizzi. Di conseguenza la configurazione più probabile del sistema è associata al massimo valore di W, visto che $\{n_i\}$ è tanto frequente quanto $W_{\{n_i\}}$ è grande. In particolare, si può dimostrare che tutte le configurazioni con W diverso dal massimo hanno una probabilità estremamente bassa di realizzarsi. Ciò è analogo a dire che un sistema all'equilibrio è disposto in uno stato che massimizza il numero possibili di microstati.

Si può dimostrare dal calcolo della configurazione massima $\{n_i*\}$ che la distribuzione di probabilità per l'ensemble canonico è:

$$P_i = \frac{e^{-\beta E_i}}{Z} \ . \tag{1.2}$$

Al denominatore il termine Z è definito *funzione di partizione* ed è dato da:

$$Z = \sum_i e^{-\beta E_i} \tag{1.3}$$

La formula 1.2 rappresenta quindi la probabilità che un sistema nell'ensemble di macrostato definito da N, V e T sia in un microstato con energia E_i. Tale probabilità è governata da $e^{-\beta E_i}$, detto *termine di Boltzmann.*

Come per l'ensemble microcanonico, anche dall'ensemble canonico è possibile ricavare le grandezze note in termodinamica. Inoltre, considerando il termine β nell'equazione precedente 1.2 e la temperatura T, si può dimostrare che le due quantità sono poste in relazione dalla seguente eguaglianza: $\beta = \frac{1}{k_B T}$.

1.1.3 Energia, entropia e temperatura

La distribuzione di probabilità dell'ensemble canonica consente di definire l'energia interna U del sistema come:

$$U = \sum_i P_i E_i = -\frac{\partial ln Z}{\partial \beta}. \tag{1.4}$$

Dall'energia interna è possibile ricondursi all'*energia libera di Helmholtz* F, definita come:

$$F = U - TS \ , \tag{1.5}$$

dove T ed S sono rispettivamente la temperatura e l'entropia del sistema. I passaggi che conducono all'equazione 1.6 identificano anche la nota formula per l'energia libera pari a:

$$F = k_B T ln \, Z \ . \tag{1.6}$$

Tutte le grandezze note in termodinamica sono calcolabili attraverso l'energia libera di Helmholtz. Tra queste l'entropia S è ottenibile da:

$$S = -\frac{\partial F}{\partial T} \ . \tag{1.7}$$

É possibile però anche definire l'entropia in relazione al numero Ω dei microstati di un sistema:

$$S = k_B \, ln \, \Omega \tag{1.8}$$

dove k_B è la costante di Boltzmann e consente la corrispondenza con le unità termodinamiche.

In base a quanto visto nella sezione precedente, si può affermare che un sistema isolato tendente all'equilibrio, evolverà massimizzando il numero di microstati possibili. In altre parole, il sistema tende a massimizzare l'entropia. In questa situazione la grandezza che governa l'equilibrio tra le parti del sistema è la temperatura, definita da: $\frac{1}{T} = \frac{\partial S}{\partial E}$.

Un risultato particolare legato alla temperatura, e quindi a β, si rileva nel caso in cui $T \to 0$ ($\beta \to \infty$): la distribuzione di probabilità dell'ensemble canonico in equazione 1.2, sotto tale condizione, è uguale ad una $\delta(E_{min}, E)$. Pertanto per $T \to 0$ il sistema occupa uniformemente lo stato a minor energia.

1.2 Introduzione alle Neural Networks

Le Neural Networks (NNs) sono apparati matematici in grado di modellizzare un processo mediante la combinazione parallela di singole unità di processing [4–6]. Una singola unità di processing svolge generalmente due compiti: calcola un valore di output in base alle quantità ricevute in input, e trasmette l'output una volta filtrato da una *funzione di*

attivazione. Questo meccanismo è ispirato al funzionamento dei neuroni reali, i quali trasmettono la combinazione dei segnali ricevuti solo se essa supera una certa soglia.

Una NN composta da una singola unità di processing è chiamata *percettrone*, ma spesso questo termine è utilizzato anche per indicare la singola unità di processing. Le NNs sono organizzate in *layer* su cui sono disposte le unità di processing. L'organizzazione dei layer assieme alle connessioni tra le unità di processing determinano la struttura di una NN. La struttura della rete unita al tipo di operazioni matematiche eseguite dalle unità di processing identificano la tipologia della NN. Il termine generale NNs rappresenta di conseguenza svariate tipologie di NNs. In questo lavoro sono trattate esclusivamente le cosidette *Neural Networks feed-forward fully-connected* e per comodità nei prossimi capitoli il termine NN sarà utilizzato per indicarne questa tipologia.

Le connessioni tra unità di processing simulano le sinapsi dei neuroni reali e a ciascuna di esse è attribuito un *peso sinaptico*. Il peso di una connessione formalizza matematicamente l'intensità del legame tra due neuroni. I pesi sinaptici sono i parametri della NNs e ne governano di conseguenza il funzionamento. Le NNs sfruttano delle regole di apprendimento (*learning rule*) per determinare i pesi sinaptici ottimali per modellizzare un processo.

Le NNs seguono quindi un meccanismo di apprendimento e fanno parte delle tecniche di *Pattern Recognition* e di *Machine Learning*.

1.2.1 Pattern Recognition e Machine Learning

Il Pattern Recognition comprende una serie di tecniche che hanno fondamentalmente l'obiettivo di determinare la natura di un campione in base alle sue caratteristiche. Si utilizza il termine *pattern* per indicare le associazioni tra le caratteristiche del campione in input e la natura assegnatagli in output.

Il Pattern Recognition tratta una vasta gamma di problemi, assimilabili a quattro tipologie: *classificazione*, *regressione*, *clustering* e *features extraction*. La *classificazione* è il problema di assegnare ogni campione in input ad una classe. La *regressione* è il problema di ricavare un modello di regressione (es. modello lineare) per i campioni analizzati. Il *clustering* è il problema di suddividere i campioni in input in gruppi significativi (*clusters*). La *features extraction* è il problema di estrarre delle caratteristiche primitive ed informative per i campioni studiati.

Esistono due tipi di approcci al Pattern Recognition: uno di tipo statistico o ed uno di tipo neurale. Il primo approccio mira ad individuare modelli statistici che descrivano la relazione tra input e output. Il secondo approccio invece punta a risolvere i problemi tramite l'utilizzo delle NNs.

Il Machine Learning è un settore delle scienze applicate che si occupa di apprendimento artificiale. La differenza tra Pattern Recognition e Machine Learning è sottile e fondamentalmente dovuta alle diverse origini, rispettivamente in ingegneria e in Computer Science. Infatti, le metodiche di Pattern Recognition hanno come conseguenza

naturale quella di essere strumento di apprendimento; mentre il Machine Learning è intrinsecamente nato per l'apprendimento. Data questa differenza formale, Pattern Recognition e Machine Learning sono spesso considerati sinomini. In questo testo sarà usato il nome Machine Learning indistintamente per entrambi.

In generale, ogni metodo di Machine Learning è sviluppato in due fasi: una fase di definizione del metodo, e una fase di valutazione delle *performance*. Esiste anche una terza fase che consiste nella determinazione dei migliori *iper-parametri*. Gli iper-parametri sono le quantità coinvolte nel metodo sviluppato che non sono determinate dal metodo stesso (es. numero di unità di processing negli hidden layer in una NN).

Il Machine Learning suddivide l'apprendimento artificiale in due campi: *supervised learning* e *unsupervised learning*.

Il *supervised learning* utilizza sia i dati in input che i dati in output attesi durante la fase di definizione del metodo. Le due fasi in questo caso sono chiamate *training* e *test*, e l'insieme dei campioni è suddiviso in un *training set* ed in un *test set*. Il metodo è definito sul training set (in una NN ciò equivale a determinare i pesi sinaptici) ed è in seguito valutato sul test set.

L'*unsupervised learning* invece utilizza nella fase di definizione del metodo solo i dati in input. In questo caso, le due fasi consistono solitamente nella definizione del modello su tutti i campioni e nella seguente valutazione del metodo in base agli output attesi per taluni.

1.2.2 Neural Networks feed-forward fully-connected

Le Neural Networks, come detto, possono essere di diverse tipologie. Una prima distinzione è data dal numero di layer che formano una NN. Innanzitutto, si definisca il layer su cui sono ricevuti in input i dati dei campioni *input layer* e il layer che invia il risultato in output *output layer*. Si definiscano inoltre tutti i layer interposti a taluni come *hidden layer*.

Una NN è detta a *multilayer* se esistono hidden layer. Gli hidden layer sono di importanza fondamentale, poiché consentono di modellizzare processi fortemente non-lineari.

Una seconda distinzione può essere fatta in base al meccanismo di trasmissione di dati tra i layer. Nel caso in cui la trasmissione dei dati sia direzionata unicamente dall'input layer all'output layer, la NN è definita *feed-forward*.

L'esempio più semplice di una NN feed-forward è il percettrone. Inoltre, il percettrone non ha hidden layer e per questa ragione una NN feed-forward a multilayer spesso è chiamata *multilayer perceptron* (MLP).

Una terza distinzione è data dalla densità di connessioni tra i layer. Due layer sono definiti *fully-connected* quando ogni unità di un layer è connessa a tutte le unità dell'altro layer. Ciò può realizzarsi tra ogni layer di una NN e in questo caso è la rete ad essere chiamata *fully-connected*.

Come detto in precedenza, in questo lavoro solo le NNs feed-forward fully-connected sono prese in considerazione. Si consideri in maniera esemplificativa la figura 1.1, che rappresenta un esempio di architettura di questa tipologia di NN.

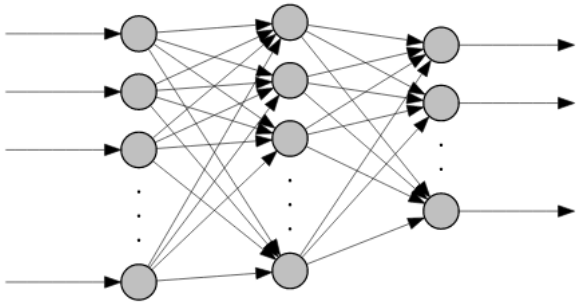

Figura 1.1: Esempio di struttura di una Neural Network con un singolo hidden layer e molteplici output. Le linee che collegano le unità di processing sono associate ad un peso sinaptico.

1.3 Introduzione alla Network Theory

La Network Theory è la branca delle scienze applicate che utilizza i grafi [3] per studiare sistemi di oggetti interagenti [7]. Un grafo è un'organizzazione di n nodi (o vertici) e di m archi. Tale organizzazione può risultare sia in un'unica struttura *connessa* che in molteplici strutture *disconnesse*. Analogamente, due nodi di un grafo possono essere sia disconnessi che connessi da uno o più archi. Alle due casistiche sono assegnati rispettivamente i termini *single-edge* e *multiedge*. Ogni nodo inoltre può potenzialmente essere connesso a tutti gli altri nodi e ciò include anche il caso in cui esso sia connesso a se stesso. L'autoconnessione di un nodo è chiamata *self-edge*. Un grafo privo di multiedges e self-edges è definito *grafo semplice*, mentre un grafo dotato di multiedges è definito *multigrafo*.

Gli archi di un grafo possono invece descrivere un tipo di relazione che avviene o in un solo verso o in entrambi i versi. I grafi caratterizzati dal primo tipo di archi sono detti *direzionati*, mentre i grafi caratterizzati dal secondo tipo sono detti *adirezionati*. Una qualsiasi sequenza consecutiva di nodi percorsa lungo degli archi definisce un *path*. Un path che ha inizio in un vertice e termina in un altro vertice ha lunghezza pari al numero totale di archi percorsi lungo il path. La distanza tra due nodi può essere quindi intesa come la lunghezza del path più corto tra i due nodi. A questo punto, è possibile definire il *diametro* di un grafo come il valore massimo fra tutte le distanze calcolabili

[3] Si noti che il termine grafo è stato coniato negli studi di matematica, mentre in Computer Science si utilizza il termine *network*.

in un grafo. In particolare, si può dimostrare che il diametro di un grafo è generalmente proporzionale al $ln\, n$.

Un'altra caratteristica osservabile grazie ai path è la presenza o meno di *loops* in un grafo. Un loop equivale ad un path che ha capo e coda in un vertice del grafo[4]. Grafi privi di loops sono definiti *aciclici*.

Un grafo connesso, adirezionato e aciclico è definito *tree* o grafo ad albero. Questa tipologia di grafi sono solitamente disegnati verticalmente e per essi si riconosce una base, i cui nodi sono chiamati *leaves*, ed una cima.

In questo lavoro sono di particolare interesse i cosiddetti *grafi bipartiti*.

1.3.1 Grafi bipartiti

Un grafo è detto bipartito se è possibile distinguere due insiemi a cui appartengono distintamente gli estremi di ciascun arco presente. In altre parole, in un grafo bipartito si riconoscono due gruppi di nodi tali per cui i nodi di un gruppo possono connettersi solo ai nodi dell'altro gruppo.

Si definisce *grafo fattoriale* un grafo bipartito che consente di descrivere funzioni "globali" multivariate in termini del prodotto di funzioni "locali" più semplici. In un grafo di tale genere i due gruppi sono: un gruppo di nodi che contiene le variabili ed un gruppo di nodi che contiene le funzioni "locali". I nodi del primo gruppo sono chiamati *nodi variabile*, mentre i nodi del secondo sono detti *nodi fattori*.

La struttura del grafo fattoriale consente di visualizzare quali sono gli argomenti delle funzioni e di quante funzioni una variabile è argomento. Un esempio di grafo fattoriale è esposto in figura 1.2 per la funzione:

$$f(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = f_1(x_1, x_2, x_3, x_4, x_6) \cdot f_2(x_3, x_4, x_7) \cdot f_3(x_4, x_5, x_7).$$

Come si può vedere dal grafo, esistono due gruppi ben distinti. Inoltre, ogni nodo variabile è connesso ad un nodo fattore solo se la variabile stessa è argomento della funzione.

Nel caso in cui tutti i nodi variabile non condividano a due a due più di un nodo fattore, il grafo fattoriale assume una struttura ad albero. Ad esempio, si supponga di rimuovere le connessioni (x_3, f_1) e (x_7, f_2) dal grafo precedente. La funzione diventa:

$$f(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = f_1(x_1, x_2, x_4, x_6) \cdot f_2(x_3, x_4) \cdot f_3(x_4, x_5, x_7).$$

e il grafo, in figura 1.3, può essere rappresentato come un tree.

[4]Di conseguenza un self-edge è considerato un loop, in particolare è detto *self-loop*.

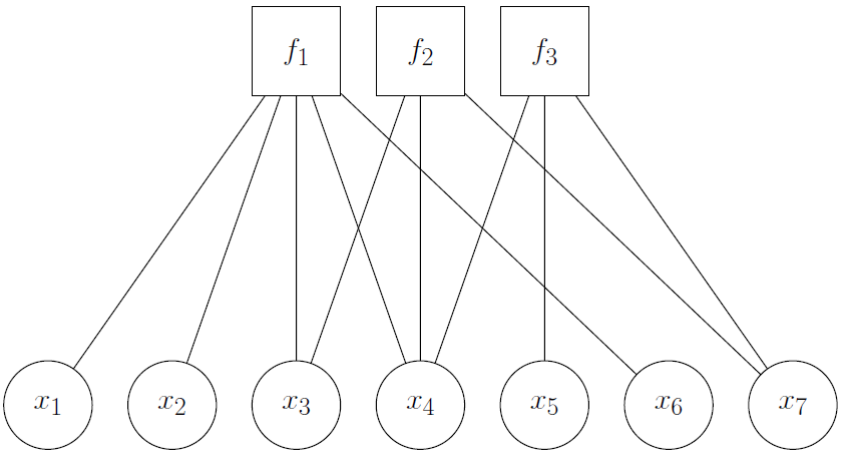

Figura 1.2: Esempio di grafo fattoriale.

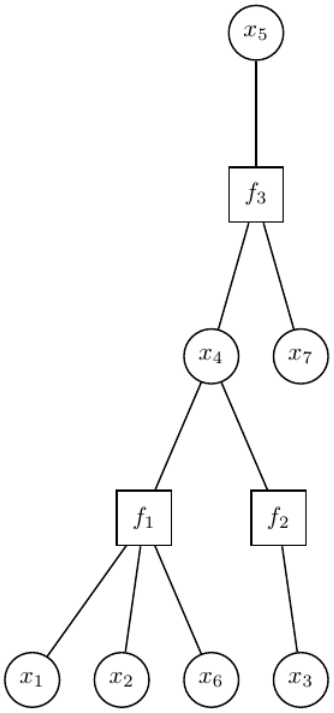

Figura 1.3: Esempio di grafo fattoriale ad albero.

1.4 Elementi di complessità computazionale

In *Computer Science* la *Teoria della complessità computazionale* studia le caratteristiche di complessità dei *problemi computazionali*. I problemi computazionali sono problemi teoricamente risolvibili da un computer attraverso l'esecuzione di un algoritmo.

Le prestazioni di un algoritmo sono valutate in termini di *complessità temporale* e di *complessità spaziale*. L'unica di queste presa in considerazione in questo lavoro è la complessità temporale.

La complessità temporale di un algoritmo è stimata a partire dal conteggio del numero di operazioni svolte, assumendo che ogni singola operazione impieghi un certo tempo[5]. Tale complessità deve essere indicativa del fatto che per dati diversi in input, ma con stessa dimensione, il numero di operazioni possa variare. In aggiunta, deve considerare il fatto che per dimensioni crescenti dei dati in input, il numero di operazioni tende ad aumentare. La dimensione dei dati in input è spesso intesa come *dimensione del problema*.

Tipicamente la complessità temporale equivale alla risoluzione peggiore possibile del problema. Una volta fissata la dimensione, ciò significa che il problema è risolto con il massimo numero possibile di operazioni. Per evidenziare la dipendenza dalla dimensione del problema, la complessità temporale indica, attraverso la notazione *O-grande*, l'andamento del numero di operazioni al crescere di tale dimensione.

Tra i problemi computazionali sono di grande rilievo i problemi decisionali, ovvero quella classe di problemi che hanno lo scopo di rispondere alle domande con un'affermazione o con una negazione. I problemi decisionali sono suddivisivi prevalentemente in problemi risolvibili in un tempo polinomiale (deterministici), e in problemi il cui risultato è verificabile come soluzione in un tempo polinomiale (non deterministici). Queste due classi di problemi sono rispettivamente chiamate *P* ed *NP*.

I problemi NP sono di grande interesse e sono utilizzati come riferimento per definire altre classi di complessità. Si definisce infatti la classe *NP-hard* come l'insieme dei problemi computazionalmente complessi almeno quanto i più complessi problemi *NP*. Ciò implica che se un problema NP-hard è risolto, il meccanismo di risoluzione sfruttato consente di risolvere tutti problemi NP. Un problema è definito perciò NP-hard se ogni problema NP è ad esso riducibile in tempi polinomiali. [6]

I problemi NP-hard che sono contemporaneamente anche NP definiscono la classe dei problemi *NP-completi*. I problemi NP-completi sono conosciuti per il rapido aumento di complessità all'aumento delle dimensioni del problema. Il crescere della complessità rende il problema progressivamente intrattabile. Un problema è definito intrattabile

[5] Il tempo di esecuzione reale di una singola operazione dipende necessariamente dalle carettistiche hardware del computer utilizzato. Per questo motivo è più efficace indicare il tempo come il numero di operazioni svolte.

[6] Si noti che l'espressione "riducibile in tempi polinomiali" significa che esiste una funzione ad andamento polimoniale che permette di riadattare un problema ad un altro.

quando, pur teoricamente risolvibile, coinvolge nella computazione una così massiccia quantità di risorse da non rendere più utile l'uso del computer.

I problemi con complessità esponenziale sono considerati intrattabili, al contrario dei problemi con complessità polinomiale ($O(N^c)$, c costante) che sono ampiamente trattabili. La maggior parte dei problemi NP-hard ed NP-completi sono tipicamente caratterizzati da andamenti esponenziali e per questo motivo sono ritenuti intrattabili.

Esiste anche una fascia intermedia di problemi considerati sufficientemente trattabili, detti sub-esponenziali, che hanno complessità inferiore all'esponenziale ma superiore alla polinomiale.

Una classe più generale di problemi utilizzata per affrontare i problemi decisionali è quella dei *Constraint Satisfaction Problems* (CSPs). I CSPs sono problemi che richiedono di trovare un gruppo di variabili sulle quali sono imposte condizioni da rispettare.

Una vasta gamma di problemi che derivano dai CSPs sono i problemi SAT (*Satisfiability problems*). I problemi SAT equivalgono ai CSPs per cui è dimostrata l'esistenza di soluzioni. Questi problemi sono tipicamente *NP-completi*.

Nel corso di questo lavoro è indicato come il training di una Neural Networks possa essere interpretato nei termini di un problema di un SAT e risulti NP-completo.

Capitolo 2

Belief Propagation Standard

L'algoritmo efficace rfBP, oggetto di questo lavoro (capitolo 5), è fondato sulle equazioni dell'algoritmo standard di Belief Propagation ed in questo capitolo sono spiegati il funzionamento e gli aspetti più interessanti legati ad esso.

Nella prima sezione sono introdotte brevemente il significato di *standard* e le origini dell'algoritmo.

Nella seconda sezione è descritta la struttura matematica di base e il funzionamento dell'algoritmo, valutando inoltre diversi approcci.

Nella terza sezione è posta in evidenza la relazione tra le funzioni usate in BP e le distribuzioni di probabilità.

In conclusione, nella quarta sezione, è proposta una reinterpretazione di Meccanica Statistica dell'algoritmo che consente di giustificare l'applicazione della BP su base fisica.

2.1 Presentazione e origini

La Belief Propagation (BP) [7–10] è un algoritmo per il calcolo di probabilità marginali che appartiene ad una serie di tecniche di *message-passing* provenienti dalla Teoria dell'Informazione [10, 11].

La Teoria dell'Informazione è una branca delle scienze matematiche che studia la trasmissione di contenuti informativi, avvalendosi sia di tecniche di statistica che di strutture matematiche come i grafi (sezione 1.3).

Gli algoritmi di message-passing sono metodi che risolvono un problema in base allo scambio di quantità chiamate *messaggi*. Dato un sistema i cui stati sono descritti da un set di variabili, questi algoritmi cercano di risolvere prevalentemente due problemi: la determinazione dello stato globale più probabile, e il calcolo delle probabilità marginali in modo da trarre informazioni locali sugli stati.

L'importanza delle probabilità marginali, o semplicemente *marginali*, è dovuta alla capacità di poter produrre informazioni su un determinato elemento, trascurando ogni conoscenza diretta con esso, ma analizzando l'ambiente in cui si ritrova.

2.2 BP standard

Quando si parla di algoritmo di Belief Propagation standard (BP) si intende un insieme di algoritmi utilizzati in diversi campi delle scienze accomunati dallo stesso meccanismo e dallo stesso scopo: il calcolo di marginali.

L'algoritmo di BP standard è sviluppato sui grafi fattoriali e sullo scambio di messaggi tra i nodi.

2.2.1 Grafi fattoriali ed expression trees

Si consideri $\{x_0, x_1, .., x_{n-1}\}$, o $\vec{x}$, un insieme di variabili con rispettivi domini $\{D_0, D_1, .., D_{n-1}\}$ e una qualsiasi funzione reale $g(\vec{x})$ per cui sia ben definita la sommatoria in $\mathbb{R}$. Per questa funzione esistono un numero n di funzioni marginali del tipo $g_i(x_i)$ che si ottengono come:

$$g_i(x_i) = \sum_{\sim x_i} g(\vec{x}), \tag{2.1}$$

dove il termine $\sim x_i$ indica che la sommatoria è eseguita su ciascun valore assumibile dagli elementi di $\{x_0, x_1, .., x_n\}$, eccetto x_i.

Sia inoltre $g(\vec{x})$ fattorizzabile nel prodotto di funzioni $f_a(\vec{X}_a)$, in cui $\vec{X}_a$ è un qualsiasi subset di $\vec{x}$ [1]. La funzione è pertanto descrivibile attraverso un grafo fattoriale che esprime $g(\vec{x})$ nella forma:

$$g(\vec{x}) = \prod_a f_a(\vec{X}_a). \tag{2.2}$$

Tale grafo fattoriale contiene un nodo *variabile* per ogni variabile x_i e un nodo *fattore* per ogni funzione $f_a(\vec{X}_a)$. Per comodità, si definiscano anche ∂x_i il set di indici dei nodi fattori f_a connessi al nodo x_i e $n(f_a)$ il set di indici dei nodi variabili x_i connessi a f_a (ovvero gli indici degli argomenti $\vec{X}_a$ della funzione).

Una proprietà importante dei grafi fattoriali è la predisposizione al calcolo delle funzioni marginali, dovuta ad un parallelismo con i cosidetti *expression trees*. Gli expression trees [7] sono strutture matematiche che rappresentano espressioni aritmetiche. In altre parole, data un'espressione aritmetica, l'expression tree corrispondente traccia le operazioni aritmetiche che la compongono (esempio in figura 2.1). Al contrario, dato un expression tree, è possibile ricostruire un procedimento di calcolo per risolvere l'espressione.

[1] Si noti che non è richiesto che i vari subsets $\vec{X}_a$ siano tra loro disgiunti.

Si dimostra che, convertendo opportunamente i grafi fattoriali in expression trees, è possibile delineare un procedimento di calcolo per i marginali. Si definiscano quindi i messaggi dai nodi variabile x_i ai nodi fattore f_a come $\nu_{x_i \to f_a}$ e i messaggi dai nodi fattore ai nodi variabile come $\hat{\nu}_{f_a \to x_i}$. L'opportuna conversione da grafo fattoriale a expression tree fissa le seguenti regole (per approfondimenti vedere l'Appendice A). Ogni nodo variabile forma ed invia un messaggio uguale a:

$$\nu_{x_i \to f_a}(x_i) = \prod_{k \in \partial x_i \setminus a} \hat{\nu}_{f_k \to x_i}(x_i) \ . \qquad (2.3)$$

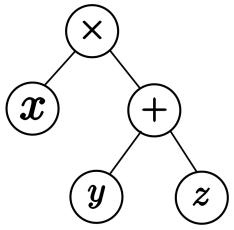

Figura 2.1: Esempio di expression tree per l'espressione artimetica $x(y + z)$.

Diversamente, ogni nodo fattore crea e invia un messaggio pari a:

$$\hat{\nu}_{f_a \to x_i}(x_i) = \sum_{\sim x_i} [f_a(\vec{X}_a) \prod_{j \in n(f_a) \setminus i} \nu_{x_j \to f_a}(x_j)] \ . \qquad (2.4)$$

In altri termini, il messaggio inviato da un nodo variabile è un prodotto parziale dei messaggi ricevuti. Invece, il messaggio inviato da un nodo fattore è il risultato di una sommatoria eseguita su un prodotto parziale dei messaggi ricevuti moltiplicato per la funzione associata al nodo fattore stesso. La figura 2.2 mostra rispettivamente un nodo

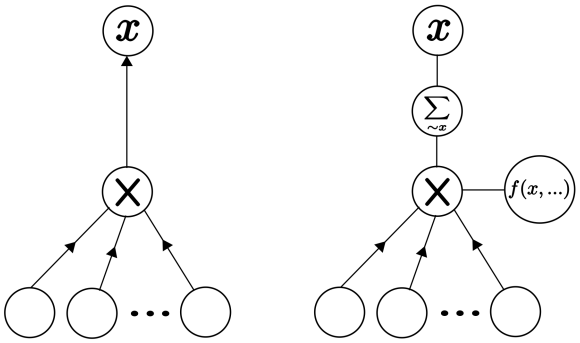

Figura 2.2: Sono rappresentati rispettivamente da sinistra a destra un nodo variabile ed un nodo fattore nella forma di expression tree.

variabile e un nodo fattore come nodi di un expression tree. Si noti che solitamente non si traccia esplicitamente l'expression tree corrispondente ma si agisce direttamente sul grafo fattoriale.

2.2.2 Algoritmo di BP standard

Ora, si assuma che la funzione $g(\vec{x})$ sia rappresentabile attraverso un grafo fattoriale ad albero (tree) [2] . Per chiarezza si considerino le seguenti equivalenze: $x_0 \equiv x$, $g_0(x_0) \equiv g(x)$ e $f_0(\vec{X}_0) \equiv f(\vec{X})$.

Si fissi come obiettivo il calcolo del marginale $g(x)$ e si prenda quindi come riferimento il nodo variabile x ponendolo in cima al tree (esempio in figura 2.3).

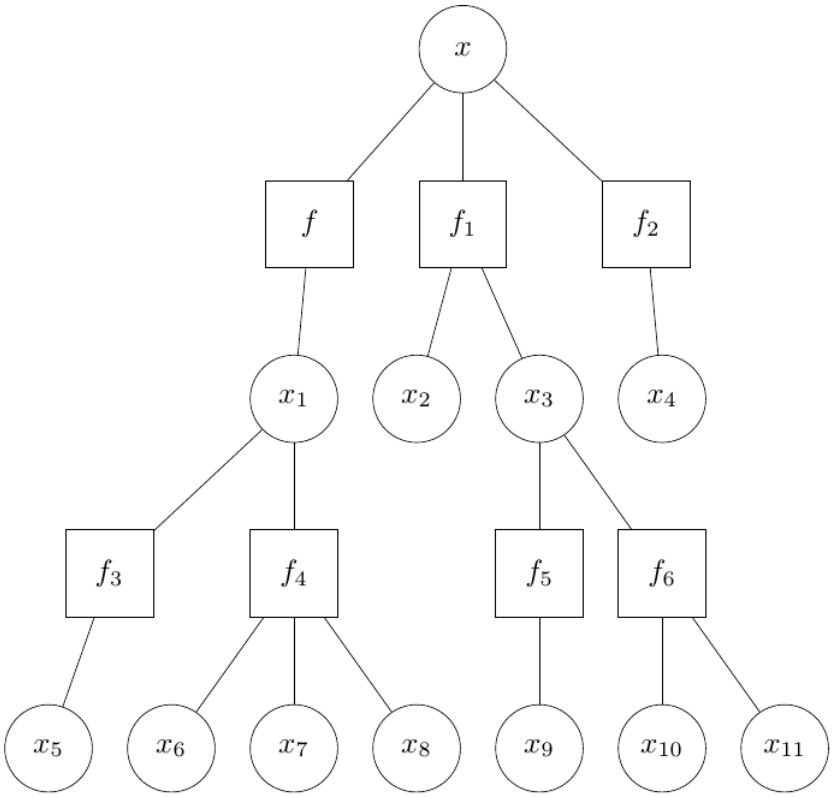

Figura 2.3: Esempio di grafo fattoriale ad albero per $g(\vec{x})$ con riferimento in x.

In base a quanto detto nella sezione precedente, assegnando implicitamente a ciascun nodo opportune operazioni aritmetiche è possibile descrivere un procedimento di calcolo per tale marginale. Ora, si segua una procedura *bottom-up*, che inizi la trasmissione dei messaggi dalle *leaves* e si concluda in cima. I messaggi si propagano dall'alto verso il basso e una volta raggiunta la cima il marginale è calcolato. Essendo in cima, il nodo

[2]Per semplificare la spiegazione, e senza perdere generalità, nelle figure mostrate si prenderà come riferimento la funzione:

$$g(\vec{x}) = f_0(x_0, x_1) f_1(x_0, x_2, x_3) f_2(x_0, x_4) f_3(x_1, x_5) f_4(x_1, x_6, x_7, x_8) f_5(x_3, x_9) f_6(x_3, x_{10}, x_{11}) \ . \quad (2.5)$$

apice non invia messaggi ma calcola semplicemente il marginale come il prodotto di tutti i messaggi ricevuti:

$$g(x) = \prod_{k \in \partial x} \hat{\nu}_{f_k \to x}(x) \ . \tag{2.6}$$

Ora, si studi il caso di voler calcolare più marginali dallo stesso grafo senza dover cambiare mano a mano la variabile di riferimento. Si valuti nuovamente il calcolo del marginale $g(x)$. In questa circostanza non è più conveniente una struttura ad albero (*non-tree*), dato che, non essendoci versi privilegiate, ciascun nodo è liberamente interpretabile sia come mittente che come ricevente (esempio in figura 2.4).

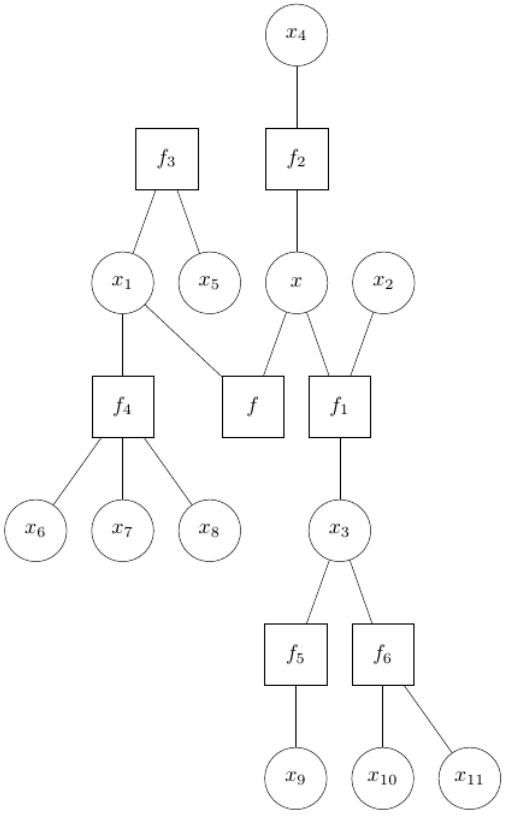

Figura 2.4: Grafo fattoriale non-tree per $g(\vec{x})$ associato all'esempio in figura 2.3.

Siccome è stata modificata solo la configurazione del grafo e non il grafo stesso, il marginale $g(x)$ è sempre calcolabile dall'equazione 2.6. Questa volta però, non è possibile seguire un percorso unidirezionale, come con la procedura bottom-up, e il nodo *variabile* può inviare dei messaggi.

A questo punto, si consideri senza perdere generalità la coppia di nodi connessi $\{x, f\}$ (ricordando $f = f(\vec{X}) \equiv f_0(\vec{X}_0)$) e si imposti la trasmissione dei messaggi in modo che

essa sia orientata su questa coppia di nodi. In questa situazione il nodo x invia un messaggio $\nu_{x\to f}$ ad f, pari al prodotto dei messaggi ricevuti da x provenienti da tutti i nodi fattori eccetto f (eq. 2.3). Il confronto di questo messaggio con la formula per il marginale 2.6, suggerisce che quest'ultimo si possa ricavare nuovamente come:

$$g(x) = \nu_{x\to f} \cdot \hat{\nu}_{f\to x} \ . \tag{2.7}$$

In altre parole, mentre nel grafo fattoriale a tree la procedura terminava in cima per assenza di nuovi messaggi da inviare, nel grafo fattoriale non-tree il calcolo del marginale si realizza nell'interazione tra due messaggi di natura differente. Questo meccanismo è valido per il calcolo di qualsiasi altro marginale oltre a $g(x)$ senza dover modificare la configurazione di esso.

L'utilizzo del grafo non-tree garantisce quindi il calcolo di tutti i marginali sulla stessa rappresentazione. Nonostante ciò, i marginali possono essere calcolati solo uno dopo l'altro.

Si supponga ora di voler determinare simultaneamente tutti i marginali. Questo problema implica direttamente che non è conveniente stabilire una direzione e che è necessario applicare una procedura indipendente dai vari percorsi all'interno del grafo.

Si dimostra che il problema è risolvibile grazie ad una procedura di aggiornamento dei messaggi inviati dai nodi fattori ai nodi variabile e viceversa. Questa procedura necessita della definizione di una sorta di tempo t che indicizzi i progressivi aggiornamenti, e di una condizione iniziale (ovvero per $t = 0$) su cui poter avviare tali aggiornamenti [3]. Si riscrivano i messaggi definiti nelle equazioni 2.3 e 2.4 rispettivamente come:

$$\nu^t_{x_i\to f_a}(x_i) = \prod_{k\in\partial x_i\setminus a} \hat{\nu}^t_{f_k\to x_i}(x_i), \tag{2.8}$$

$$\hat{\nu}^{t+1}_{f_a\to x_i}(x_i) = \sum_{\sim x_i}[f_a(\vec{X}_a) \prod_{j\in n(f_a)\setminus i} \nu^t_{x_j\to f_a}(x_j)], \tag{2.9}$$

Le equazioni 2.8 e 2.9 sono eseguite alternativamente nel corso delle iterazioni e ad ogni iterazione l'aggiornamento è effettuato contemporaneamente su tutti i messaggi corrispondenti. In altre parole, quando è svolta l'equazione 2.8, questa è eseguita su ogni messaggio da nodo variabile a nodo fattore allo stesso t. Allo step successivo $t + 1$ è svolta invece l'equazione 2.9 anch'essa per ciascun messaggio da nodo fattore a nodo variabile.

Queste equazioni di aggiornamento sono alternate per un numero di iterazioni prestabilito o tale per cui si abbia la convergenza dei messaggi. Un set di messaggi a cui si

[3] Si noti che una condizione iniziale tipica è

$$\nu^0_{x_i\to f_a} = 1 \ \forall i, a.$$

converga durante il corso degli aggiornamenti è detto *punto fisso* e per tali messaggi si usa la notazione priva del parametro t: $\nu_{x_i \to f_a}$ e $\hat{\nu}_{f_a \to x_i}$.

Un punto fisso è detto *stabile* se durante il corso degli aggiornamenti i messaggi intermedi appartengono sempre ad un suo intorno. Formalmente ciò può essere scritto nel caso di $\nu_{x_i \to f_a}$ come:

$$|\nu_{x_i \to f_a} - \nu^t_{x_i \to f_a}| < \epsilon \quad \forall i, a, \tag{2.10}$$

in cui ϵ è la distanza che definisce l'intorno. Se inoltre si ha:

$$\lim_{t \to \infty} |\nu_{x_i \to f_a} - \nu^t_{x_i \to f_a}| \to 0 \quad \forall i, a \tag{2.11}$$

il punto fisso è anche *attrattivo*.

Il set di equazioni 2.8 e 2.9 definisce una *regola di aggiornamento* dei messaggi che la figura 2.5 consente di visualizzare.

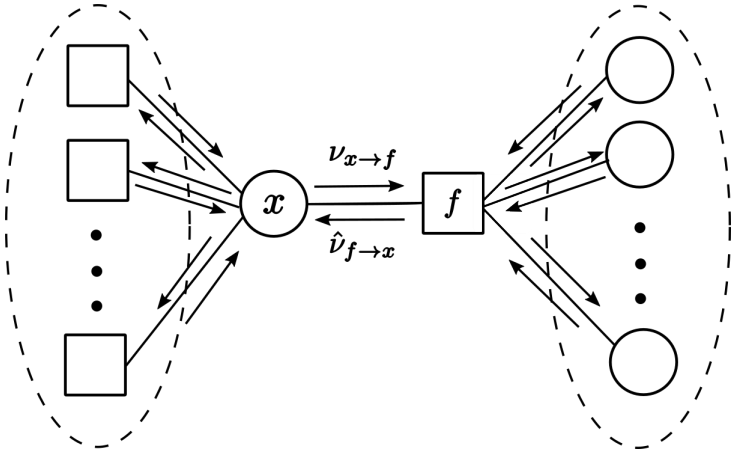

Figura 2.5: Rappresentazione di un grafo bipartito su cui è eseguita la regola di aggiornamento dei messaggi di BP.

Una volta ottenuto un punto fisso, è possibile sfruttare i messaggi come in precedenza (eq. 2.6) per calcolare i marginali:

$$g_i(x_i) = \prod_{k \in \partial x_i} \hat{\nu}_{f_k \to x_i}(x_i). \tag{2.12}$$

Non è necessario che il grafo fattorale sia rappresentabile ad albero per sfruttare la regola di aggiornamento, ma, nel caso in cui lo sia, è possibile dimostrare che dopo un certo numero di step i messaggi convergono e che i marginali ritrovati sono esatti (Appendice B).

Per un grafo fattoriale generale invece, le funzioni ricavate da 2.12 sono tipicamente solo approssimazioni dei marginali esatti, anche nel caso di convergenza dei messaggi. Per questo, si preferisce attribuire comunemente a tali funzioni l'appellativo *belief*.

La procedura mostrata in questa sezione è chiamata generalmente *algoritmo di somma-prodotto*[4](visto il coinvolgimento di tali operazioni aritmetiche), ma ad esso è attribuito anche il nome di *Belief Propagation* (BP). Sinteticamente, si può affermare che l'algoritmo di BP è in grado di semplificare la complessità del calcolo dei marginali attraverso una sorta di riorganizzazione delle operazioni coinvolte.

Per completezza, si valuti il caso in cui il destinatario finale dei messaggi non sia uno dei nodi variabile ma uno dei nodi fattore. In questo caso, riproducendo l'algoritmo di BP, la funzione ricercata è il marginale di $g(\vec{x})$ sul set di variabili di cui il nodo fattore considerato è funzione. Perciò, dato $f_a(\vec{X}_a)$ appartenente al set di funzioni che fattorizzano $g(\vec{x})$, il marginale ricercato è:

$$g_a(\vec{X}_a) = \sum_{\sim \vec{X}_a} g(\vec{x}), \tag{2.13}$$

che è ottenuto attraverso la Belief Propagation dall'equazione:

$$g_a(\vec{X}_a) = f_a(\vec{X}_a) \prod_{j \in n(f_a)} \nu_{x_j \to f_a}(x_j) \equiv f_a(\vec{X}_a) \prod_{j \in n(f_a)} \prod_{k \in \partial x_j \setminus a} \hat{\nu}_{f_k \to x_j}(x_j). \tag{2.14}$$

2.3 Relazione con le probabilità

2.3.1 Interpretazione generale

La funzione $g(\vec{x})$ ha carattere piuttosto generale e nulla vieta che possa rappresentare una distribuzione di probabilità multivariata. Dall'ipotesi di fattorizzazione (equazione 2.2), è possibile quindi descrivere la distribuzione di probabilità generica $P(\vec{x})$ in forma:

$$P(\vec{x}) = \prod_a f_a(\vec{X}_a), \tag{2.15}$$

per la quale le funzioni $f_a(\vec{X}_a)$ sono supposte positive e finite, in modo da rendere il prodotto ben definito.

Nonostante questa interpretazione, le distribuzioni di probabilità più interessanti da studiare sono quelle normalizzate, che si ritrovano nella forma:

$$P(\vec{x}) = \frac{1}{Z} \prod_a f_a(\vec{X}_a) \tag{2.16}$$

dove Z è la funzione di partizione: $Z = \sum_{\vec{x}} \prod_a f_a(\vec{X}_a)$. É indispensabile sottolineare la presenza della funzione di partizione, visto che l'algoritmo di BP permette il calcolo

[4]Si noti in particolare che esiste un altro algoritmo di Belief Propagation detto *max-prodotto* che ha un meccanismo analogo e la cui regola di aggiornamento è la stessa della somma-prodotto se non che la sommatoria è sostituita dalla ricerca del termine massimo.

delle funzioni di probabilità marginali non normalizzate. É possibile quindi considerare l'esistenza di una funzione marginale non normalizzata $g_i(x_i) = \sum_{\sim x_i} g(\vec{x})$ e di una normalizzata $P_i(x_i) = \sum_{\sim x_i} P(\vec{x})$, messe in relazione dal termine di normalizzazione Z.

Considerando l'espressione 2.2 per g e l'espressione 2.16 per P, la funzione di probabilità marginale $P_i(x_i)$ si riduce semplicemente a:

$$P_i(x_i) = \frac{g_i(x_i)}{Z} \quad \text{dove} \quad Z = \sum_{\{x_i\}_{\forall i}} g_i(x_i). \tag{2.17}$$

I marginali $g_i(x_i)$ sono calcolati dall'algoritmo di BP e, come detto in precedenza, nel caso in cui il grafo fattoriale sia privo di cicli, il termine $g_i(x_i)$ ricavato dall'algoritmo è il marginale esatto .

Generalmente è però più conveniente chiamare il risultato dell'algoritmo $b_i(x_i)$ (*belief*) piuttosto che $g_i(x_i)$ e da caso a caso valutare se l'approssimazione è esatta o meno. Richiamando la formula 2.12 e trascurando la normalizzazione, le *belief* si ricavano da:

$$b_i(x_i) \propto \prod_{k \in \partial x_i} \hat{\nu}_{f_k \to x_i}(x_i). \tag{2.18}$$

Si noti che allo stesso modo con cui $b_i(x_i)$, una volta normalizzato, approssima $P_i(x_i)$, si definiscono anche le *beliefs* $b_a(\vec{X}_a)$ come approssimazione delle distribuzioni $P_a(\vec{X}_a)$ (equivalenti a $g_a(\vec{X}_a)$, eq. 2.14):

$$b_a(\vec{X}_a) \propto f_a(\vec{X}_a) \prod_{j \in n(f_a)} \nu_{x_j \to f_a}(x_j). \tag{2.19}$$

2.3.2 Significato dei messaggi

Si consideri il marginale $b_i(x_i)$, che, una volta normalizzato, approssima $P_i(x_i)$, e si analizzino le relazioni tra questo e i messaggi tra i nodi. Per comodità si riportano nuovamente le equazioni dei messaggi una volta raggiunto un punto fisso:

$$\nu_{x_i \to f_a}(x_i) = \prod_{k \in \partial x_i \setminus a} \hat{\nu}_{f_k \to x_i}(x_i), \tag{2.20}$$

$$\hat{\nu}_{f_a \to x_i}(x_i) = \sum_{\sim x_i} [f_a(\vec{X}_a) \prod_{j \in n(f_a) \setminus i} \nu_{x_j \to f_a}(x_j)]. \tag{2.21}$$

Il messaggio da nodo variabile a nodo fattore $\nu_{x_i \to f_a}$ rappresenta un marginale cavo di x_i poichè, pur conservando la stessa forma di $b_i(x_i)$ (eq. 2.18) trascura uno dei nodi fattori (f_a in questo caso). Perciò $\nu_{x_i \to f_a}$ può essere anche riscritto come una distribuzione $b_{i,\{f_{k \neq a}\}}(x_i)$, dove $\{f_{k \neq a}\}$ indica che tutti i nodi fattori $f_k \in \partial x_i$ sono coinvolti tranne f_a.

Il messaggio $\hat{\nu}_{f_a \to x_i}$ trasmesso da nodo fattore a nodo variabile, invece, quantifica il contributo dato dal nodo fattore f_a alla *belief* $b_i(x_i)$. In particolare, il messaggio equivale

alla marginalizzazione rispetto a x_i del marginale cavo $b_{a,\{x_{j\neq i}\}}(\vec{X}_a)$ (eq. 2.19), dove $\{x_{j\neq i}\}$ indica che nella produttoria è trascurato il nodo x_i. In più, siccome l'equazione contiene termini che dipendono dai nodi fattori $\{f_{k\neq a}\}$, il messaggio può essere anche espresso come $b_{a,\{f_{k\neq a}\}}(x_i)$.

2.4 Giustificazione fisica e statistica della BP

Riprendendo gli argomenti del capitolo 1.1.2, per un sistema di n particelle con stati $\{x_1, x_2, ..., x_n\}$, a temperatura T e volume V, la probabilità di ottenere all'equilibrio termico un certo stato $\vec{x}$ è descritta da:

$$P(\vec{x}) = \frac{e^{-\frac{E(\vec{x})}{k_B T}}}{Z} \quad \text{, dove} \quad Z = \sum_{\forall \vec{x}} e^{-\frac{E(\vec{x})}{k_B T}}. \tag{2.22}$$

La formula 2.22 deriva dall'equazione precedente 1.2 e l'energia $E(\vec{x})$ è l'energia associata al sistema nello stato $\vec{x}$.

Trascurando la situazione fisica e supponendo di considerare inizialmente una certa distribuzione di probabilità $P(\vec{x})$, la formula 2.22 può essere utilizzata per poter definire un'energia per il sistema. Tramite questo approccio, il termine $k_B T$ diventa fondamentalmente una costante utile ad impostare una scala energetica e quindi per comodità, e senza perdere generalità, può essere fissato ad 1.

Si consideri pertanto la distribuzione di probabilità definita in 2.16 e la si eguagli alla prima espressione in 2.22. Dopo alcuni passaggi algebrici si ricava un'energia associata al sistema:

$$E(\vec{x}) = -\sum_a ln(f_a(\vec{X}_a)). \tag{2.23}$$

Ora, come citato nel capitolo 1.1, le quantità di termodinamica possono essere ricavate a partire dall'energia libera di Helmholtz (equazione 1.6). Avendo prefissato $k_B T = 1$, ovvero $\beta = 1$, l'energia libera di Helmholz diventa:

$$F = -lnZ. \tag{2.24}$$

Il calcolo di F dipende quindi in generale dal calcolo della funzione di partizione Z. Risolvere la sommatoria presente nella funzione di partizione risulta tipicamente difficile da trattare da un punto di vista analitico. Perciò, si consideri il seguente metodo alternativo. Si introducano due grandezze ausiliari, una distribuzione di probabilità $b(\vec{x})$ e la cosiddetta *energia libera variazionale* $F(b)$ di forma:

$$F(b) = U(b) - S(b), \tag{2.25}$$

in cui

$$U(b) = \sum_{\forall \vec{x}} b(\vec{x}) E(\vec{x}) \quad \text{e} \quad S(b) = -\sum_{\forall \vec{x}} b(\vec{x}) ln(b(\vec{x})). \tag{2.26}$$

I termini $U(b)$ e $S(b)$ definiscono rispettivamente l'*energia media variazionale* (analoga all'energia interna in equazione 1.4) e l'*entropia variazionale*.

Da 2.22 si può ottenere $E(\vec{x}) = -lnP(\vec{x}) - lnZ$ e combinando tale espressione e quelle contenute in 2.25 e 2.26, si può ricavare, considerando anche la condizione di normalizzazione $\sum_{\forall \vec{x}} b(\vec{x}) = 1$:

$$F(b) = F + \sum_{\forall \vec{x}} b(\vec{x}) ln\left(\frac{b(\vec{x})}{P(\vec{x})}\right). \tag{2.27}$$

Da eq. 2.27 si nota che $F(b)=F$ solo quando $b(\vec{x}) = P(\vec{x})$ e si può dimostrare che $F(b) \geq F \ \forall b$. Ciò implica che minimizzare $F(b)$ rispetto a $b(\vec{x})$ consente di ricavare direttamente F e l'annessa distribuzione di probabilità $P(\vec{x})$. Questo metodo alternativo consiste quindi nel definire una forma approssimativa per $F(b)$ e nel minimizzare tale grandezza.

Le procedure per approssimare l'energia libera variazionale sono svariate e alcune delle più efficienti sono basate sulla suddivisione di un grafo in *regioni* ben definite (esempio in figura 2.6). Per questo motivo tali approssimazioni sono caratterizzate dall'appellativo *region-based free energy*.

É possibile dimostrare che esiste un metodo di approssimazione *region-based* le cui soluzioni sulla minimizzazione dell'energia libera variazionale sono equivalenti ai risultati dati dall'algoritmo di Belief Propagation *standard* ([9]). Questo metodo è stato proposto da *Bethe* e per questo è chiamato *approssimazione di Bethe*.

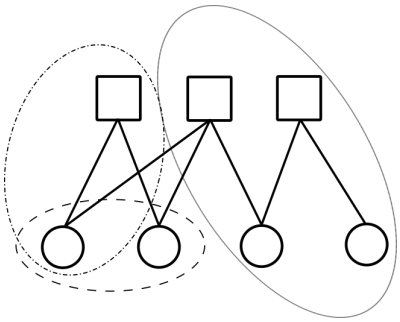

Figura 2.6: Suddivisione di un grafo fattoriale in regioni.

Nello specifico, le regioni considerate da Bethe sono costituite da un nodo fattore assieme a tutti i nodi variabile ad esso connesse (figura 2.7). Si dimostra che i punti stazionari dell'approssimazione dell'energia libera variazionale di Bethe, $F_{Bethe}(b)$, corrispondono ai punti fissi dell'algoritmo di BP, con le relative $b_i(x_i)$ uguali a eq. 2.18.

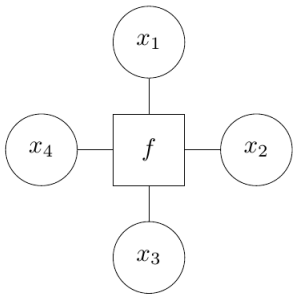

Figura 2.7: Esempio di una regione di Bethe.

Capitolo 3

Spin Glasses e Belief Propagation

Questo capitolo ha lo scopo di mostrare l'esistenza di una corrispondenza tra le equazioni che regolano la Belief Propagation e le equazioni che determinano le cosidette *magnetizzazioni cave* nel campo degli Spin Glasses. Grazie a questa corrispondenza è possibile interpretare la rete su cui è eseguito l'algoritmo del capitolo 5 come un sistema di Spin Glasses, ed è possibile comprendere le origini delle equazioni implementate in tale algoritmo.

Nelle prime due sezioni sono introdotti i caratteri generali del modello di Ising e degli Spin Glasses, soffermandosi in particolare sulle quantità più rilevanti ai fini di questo lavoro.

Nella terza sezione è infine esplicitata la corrispondenza tra gli Spin Glasses e la BP.

3.1 Introduzione al modello di Ising

Il *modello di Ising* [1–3] è un modello matematico che tenta di descrivere il carattere ferromagnetico di alcuni materiali metallici. In questi materiali gli atomi sono disposti in un reticolo periodico ed ogni atomo è dotato di uno spin. Sotto opportune condizioni si può osservare una polarizzazione spontanea di tali spin, che si riproduce macroscopicamente nella generazione di un campo magnetico.

Pertanto, si consideri un reticolo $V = \{i\}$ formato da N punti i, detti *siti*, e si associ a ciascun sito una variabile $s_i \in \{-1, 1\}$ che rappresenti lo spin i-esimo. Si definisca una generica configurazione degli spin come $\vec{S} = \{s_i\}$. La forma generale per l'Hamiltoniana del modello di Ising è espressa dalla formula:

$$H(\vec{S}; \{\epsilon_{ij}\}, \vec{h}) = -\sum_{(ij)\in B} \epsilon_{ij} s_i s_j - \sum_{i=1}^{N} h_i s_i, \tag{3.1}$$

in cui (ij) indica una coppia di spin appartenenti ad un set di coppie $B = \{(ij)\}$. Si noti che per convenzione si è scelto di anteporre e di posporre al ";" rispettivamente le

variabili e i parametri del sistema. Per favorire la chiarezza del testo, però, le future equazioni sono espresse solo in un funzione delle variabili, trascurando la parte relativa ai parametri. Perciò $H(\vec{S}; \{\epsilon_{ij}\}, \vec{h}) \equiv H(\vec{S})$.

L'insieme B dipende dal problema da risolvere ma, in generale, è composto dalle coppie di spin tra cui si assume un legame. In particolare, nel caso si considerino i legami tra coppie di atomi più vicini tra loro, si definisce (ij) come $< ij >$.

Ciascuna variabile ϵ_{ij} esprime l'interazione tra gli spin i-j, che è positiva se l'interazione è di tipo ferromagnetico e negativa se di tipo antiferromagnetico. Nel caso sia presente un campo magnetico esterno, i termini h_i rappresentano il tipo di effetto indotto su ciascun spin: h_i è positivo se lo spin tende ad allinearsi al campo magnetico esterno ed è negativo altrimenti.

Nei materiali metallici gli spin tendono a polarizzarsi spontaneamente e quindi solitamente per questi tipi di sistemi si assume che i termini ϵ_{ij} e h_i siano costanti: $\epsilon_{ij} = \epsilon$ e $h_i = h$. Ora, si supponga di studiare questo sistema di spin all'equilibrio. Come spiegato nel primo capitolo, la Meccanica Statistica studia i sistemi all'equilibrio attraverso la distribuzione di Boltzmann (eq. 1.2) [1] :

$$P(\vec{S}) = \frac{1}{Z} e^{-\beta E(\vec{S})} \quad \text{dove} \quad Z = \sum_{\vec{S}} e^{-\beta E(\vec{S})}, \tag{3.3}$$

dove l'energia totale $E(\vec{S}) \equiv H_{\vec{S}}$ [2] .

Una grandezza rilevante per caratterizzare le interazioni tra gli spin è la *magnetizzazione*, definita come:

$$m = \frac{1}{N} \langle \sum_{i=1}^{N} s_i \rangle_{\vec{S}} \equiv \frac{1}{N} \sum_{\vec{S}} \left[P(\vec{S}) \sum_{i=1}^{N} s_i \right]. \tag{3.4}$$

La magnetizzazione è un parametro d'ordine, nel senso che consente di ottenere informazioni sullo stato del sistema. In particolare, si può dimostrare che al di sopra di una certa temperatura T_C, detta *Temperatura di Curie*, la magnetizzazione è nulla; mentre al di sotto di tale soglia, $m \neq 0$. Ciò implica l'esistenza di due fasi per il sistema e che il fenomeno macroscopico si presenti solo nella fase in cui $T < T_C$.

Ricordando che dall'energia libera F è possibile ricondursi alle grandezze fisiche e che tale energia si esprime come $F = -ln\, Z$, risulta indispensabile saper ricavare la funzione di partizione [3] .

[1] Si noti che in letteratura la funzione di partizione Z è talvolta riportata nella forma:

$$Z = Tr\, e^{-\beta E} \quad \text{in cui} \quad Tr \equiv \sum_{\vec{S}}. \tag{3.2}$$

[2] In Meccanica Classica l'Hamiltoniana di un sistema coincide con l'energia totale del sistema stesso.

[3] Si noti che $F = -ln\, Z$ sempre nel caso in cui si possa assumere $k_B T = 1$, altrimenti $F = -k_B T ln\, Z$.

Teoricamente la funzione di partizione è sempre calcolabile dalla forma 3.2, ma tale procedura è impraticabile visto che il numero degli addendi cresce come 2^N. Per tale ragione, il problema è solitamente trattato mediante teorie di approssimazione.

3.1.1 Accenno alle teorie di approssimazione

La teoria di approssimazione maggiormente utilizzata è l'*approssimazione di campo medio*, che si concretizza in numerose formulazioni. L'assunto fondamentale della teoria di campo medio è l'indipendenza statistica locale tra gli spin, ovvero sono trascurate le fluttuazioni dei vari spin attorno al loro valore medio. Tra le diverse formulazioni della teoria di campo medio del modello di Ising sono presenti il *modello di Weiss*, l'*approssimazione di Bragg-Williams* e l'*approssimazione di Bethe-Peierls*.

Alcune di queste approssimazioni utilizzano un approccio di tipo variazionale (in analogia a quanto descritto in sezione 2.4), in cui per semplificare la distribuzione di probabilità $P(\vec{S})$ (eq. 3.3) si suppone inizialmente che esistano distribuzioni di probabilità locali $P_i(\sigma_i)$ e che $P(\vec{S})$ sia fattorizzabile per esse:

$$P_i(\sigma_i) = \sum_{\vec{S}} P(\vec{S})\delta(s_i, \sigma_i) \quad \text{e} \quad P(\vec{S}) \approx \prod_i P_i(s_i). \tag{3.5}$$

Queste tecniche mirano a minimizzare l'energia libera F per ricavare sia $P_i(s_i)$ che le magnetizzazioni.

Si noti anche che, in analogia alla formula 3.4, la magnetizzazione *locale* è definita da:

$$m_i = \sum_{s_i} s_i P(s_i), \tag{3.6}$$

che nel caso di variabili di Ising $s_i = \pm 1$ è compatibile con la forma generale:

$$P_i(s_i) = \frac{1 + m_i s_i}{2}. \tag{3.7}$$

Attraverso alcune procedure di approssimazione (e sotto opportune condizioni) è possibile ricondursi ad un'equazione di stato per la magnetizzazione della forma tipica:

$$m = \frac{\sum_{\vec{S}} s_i e^{-\beta E(\vec{S})}}{Z} = tanh\beta(\epsilon z m + h), \tag{3.8}$$

in cui z indica il numero di legami per sito ed è chiamato *numero di coordinazione*. Questa equazione permette di studiare il punto critico in cui avviene la transizione di fase, ovvero quando la magnetizzazione si annulla.

3.2 Introduzione agli Spin Glasses

Il modello d'interazione tra spin espresso dal modello di Ising può essere adattato anche nello studio dei cosiddetti sistemi di Spin Glasses. I sistemi di Spin Glasses sono sistemi di spin i cui termini d'interazione ϵ_{ij} sono variabili random che non evolvono nel corso del tempo. Lo stato di Spin Glass è una fase di estremo interesse in Meccanica Statistica [12–15] poichè la natura random dei termini d'interazione crea una competizione tra legami ferromagnetici e antiferromagnetici che induce gli spin a configurarsi in stati metastabili. É possibile dimostrare attraverso approssimazioni di campo medio che tale fase di Spin Glass esiste per basse temperature.

Un'analogia a questo stato della materia può essere descritto considerando la struttura di un metallo in contrapposizione alla struttura del vetro. I metalli sono contraddistinti da una struttura cristallina, mentre il vetro è caratterizzato da una struttura in cui gli atomi sono disposti casualmente nello spazio. La peculiarità del vetro, però, è dovuta al fatto che, pur avendo posizioni casuali, gli atomi del vetro rimangono fissi nel corso del tempo. I sistemi, come gli Spin Glasses, caratterizzati da parametri random costanti nel tempo sono anche definiti *sistemi disordinati*.

L'Hamiltoniana dei sistemi di Spin Glasses può essere definita, sempre per spin uguali a ± 1 (*variabili di Ising*), come:

$$H(\vec{S}) = -\sum_{(ij)\in B} \epsilon_{ij} s_i s_j, \tag{3.9}$$

in analogia all'equazione 3.1. Siccome le variabili ϵ_{ij} sono random, l'Hamiltoniana 3.9 dipende dal set di $\vec{\epsilon} = \{\epsilon_{ij}\}$ generati da una certa distribuzione di probabilità $P(\epsilon_{ij})$ e quindi anche l'energia libera F dipende da esso. Ciò suggerisce che per ricavare l'energia libera F è necessario effettuare una media rispetto alla distribuzione $P(\epsilon_{ij})$ [4] .

Tale genere di procedura è però difficile da compiere ed un modo per semplificare il calcolo è sfruttare il seguente limite:

$$\langle ln\, Z\rangle_{\{\epsilon_{ij}\}} = \lim_{n\to 0} \frac{\langle Z^n\rangle_{\{\epsilon_{ij}\}} - 1}{n}. \tag{3.10}$$

Questa procedura è chiamata *metodo delle repliche*. Infatti, sono considerate n repliche ed è calcolata la media sul prodotto delle funzioni di partizioni (Z^n), per poi ricavare il limite per $n \to 0$.

I metodi di approssimazione per affrontare i problemi sugli Spin Glasses sono innumerevoli e, come nel caso del modello di Ising, la stragrande maggioranza di essi è riconducibile a modelli di campo medio.

[4] Indicata come avvenuto in precedenza dalle parentesi angolari $\langle\rangle_{\{\epsilon_{ij}\}}$.

3.2.1 Accenni ai modelli di campo medio per gli SG

I principali modelli di campo medio utilizzati nello studio degli Spin Glasses sono il *modello di Sherrington–Kirkpatrick* (SK), i *modelli di Edwards-Anderson* e il *modello di Bethe*.

Il modello SK è di particolare importanza perchè sotto opportune condizioni gli altri modelli si riconducono ad esso. In questo modello si considerano tutti gli ϵ_{ij} casuali e non nulli rispetto ad una distribuzione Gaussiana con varianza $N^{-\frac{1}{2}}$. Seguendo il *metodo delle repliche* è possibile calcolare $\langle Z^n \rangle_{\{\epsilon_{ij}\}}$ ed ottenere una formulazione per l'energia libera che dipende da due variabili m_α e $q_{\alpha\beta}$, in cui α e β indicizzano le repliche. Si può verificare che tali variabili m_α e $q_{\alpha\beta}$, che si definiscono per comodità nel corso dei calcoli, sono dei parametri d'ordine: sono rispettivamente la magnetizzazione (vista precedentemente) e il *parametro d'ordine di Spin Glass*.

Per calcolare questi parametri e l'energia libera è necessario conoscere esplicitamente la dipendenza di m_α e $q_{\alpha\beta}$ dagli indici delle repliche. É intuitivo supporre inizialmente che tali parametri non dipendano dal tipo di repliche, dato che esse sono state introdotte per scopi tecnici. Da questa supposizione, detta *simmetria delle repliche* (RS), è possibile ricavare un'equazione di stato.

La soluzione dell'equazione di stato conduce per basse temperature ad un'energia libera associata ad un'entropia negativa. Un risultato di questo tipo, privo di alcun fondamento fisico, induce ad assumere che i parametri d'ordine m_α e $q_{\alpha\beta}$ dipendano effettivamente dalle repliche. Le teorie che si concentrano su questo aspetto del problema sono dette *teorie di rottura della simmetria* (RSB).

Un altro approccio per ottenere l'energia libera F che eviti di calcolare la media $\langle\rangle_{\{\epsilon_{ij}\}}$ si basa su un'equazione, introdotta da Thouless, Anderson e Palmer (per questo denominata *equazione di TAP*), che è soddisfatta dalla magnetizzazione locale m_i. Questa equazione si può derivare, in particolare, dal *metodo delle cavità* che è una potente strategia per conoscere le proprietà degli Spin Glasses.

Il metodo delle cavità consente di calcolare, trascurando gli effetti di campo esterno, la distribuzione di probabilità marginale $P_i(s_i)$ in funzione del campo locale $\tilde{h_i} = \sum_j \epsilon_{ij} s_j$. In dettaglio, si può ricavare:

$$P_i(s_i) \propto \int d\tilde{h_i} P(s_i, h_i) \quad \text{dove} \quad P(s_i, h_i) \propto e^{\beta \tilde{h_i} s_i} P(\tilde{h_i} \setminus s_i). \tag{3.11}$$

Il calcolo di $P_i(s_i)$ si può effettuare solo conoscendo il *campo di cavità* $P(\tilde{h_i} \setminus s_i)$, ovvero il campo locale risultante trascurando s_i dal sistema.

Per il modello SK questo campo di cavità è assunto Gaussiano sotto certe condizioni, mentre nel caso del modello di Bethe questa assunzione è sempre vera. Il modello di Bethe (analogo al modello di Bethe-Peierls citato in precedenza), in particolare, permette di tracciare una corrispondenza tra la Meccanica Statistica e l'algoritmo di Belief

Propagation. Questo fatto è in perfetta sintonia con i contenuti della sezione 2.4 , i quali accennavano alla giustificazione fisica sulle cosiddette regioni di Bethe.

3.3 Corrispondenza tra i modelli SG e l'algoritmo BP

3.3.1 Approssimazione di Bethe

Si riprenda in considerazione il modello di Ising visto nella sezione 3.1 e il risultato in equazione 3.8 per la magnetizzazione in una classica approssimazione di campo medio, che si riporta per comodità:

$$m = tanh(\beta\epsilon z m). \tag{3.12}$$

I risultati dati da questa equazione si possono migliorare sfruttando un diverso approccio.

Si consideri il caso bidimensionale e si cerchi di calcolare la magnetizzazione locale di un certo s rispetto alle magnetizzazioni vicine τ_i (con $i = 1..4$), supponendo che esse siano tra loro scorrelate. Perchè siano scorrelate, si rimuova s dal reticolo così creando una cavità circondata dai τ_i. Si supponga inoltre che i τ_i abbiano una magnetizzazione pari a m_τ.

Si può dimostrare che la magnetizzazione m dello spin s è data dalla formula:

$$m = tanh[z\, artanh(tanh(\beta\epsilon)\, m_\tau)]. \tag{3.13}$$

Per ricavare m_τ è sufficiente attuare lo stesso procedimento rimuovendo assieme a s anche uno tra i τ_i. In tal modo si ricava la magnetizzazione m_τ risolvendo l'equazione [5] :

$$m_\tau = tanh[(z-1)artanh(tanh(\beta\epsilon)m_\tau)]. \tag{3.14}$$

Ora, si studi un caso più complesso in cui i valori delle magnetizzazioni siano differenti. Si consideri un nodo i e si ricavi l'Hamiltoniana dalla forma:

$$H = \frac{1}{2}\sum_i \sum_{j\in\partial i} \epsilon_{ij} s_i s_j, \tag{3.15}$$

in cui ∂i rappresenta l'insieme dei nodi connessi a i e il termine $\frac{1}{2}$ consente di non contare due volte lo stesso legame.

Si supponga che esista uno stato del sistema γ che esista sia per N spin che per $N-1$ spin, nel quale è rimosso un certo spin i. Come in precedenza, è ragionevole supporre che due spin qualsiasi siano scorrelati tra loro, vista la loro lontananza, e per questo la

[5] Questo tipo di approssimazione è adatta ai cosiddetti grafi random poichè una volta rimosso un nodo, i nodi circostanti si ritrovano ad una distanza tale da poter ritenere $\langle \tau_{i_a}, \tau_{i_b} \rangle_{\{\epsilon_{ij}\}} \approx m_a m_b$.

distribuzione di probabilità di essi possa essere fattorizzata. Questo aspetto, come visto in precedenza, consente di descrivere la distribuzione di probabilità in funzione delle magnetizzazioni.

Analogamente al procedimento che ha condotto alle formule 3.13 e 3.14, è possibile dimostrare che la magnetizzazione di uno spin i in assenza di $a \in \partial i$ ($m_{i;\sim a}$) è data da:

$$m_{i;\sim a} = tanh\left[\sum_{j \in \partial i \setminus a} artanh(tanh(\beta \epsilon_{ij}) m_{j;\sim i})\right]. \tag{3.16}$$

Da questo genere di procedura sono generate un numero di equazioni pari alla moltiplicazione Nz tra il numero di spin e il numero di spin vicini.

Le magnetizzazioni ottenute dalle formula 3.16 non sono però le magnetizzazioni *vere*, poichè è presente una cavità; per questo sono dette *magnetizzazioni cave*. Una volta risolte le equazioni per le magnetizzazioni cave è possibile determinare le magnetizzazioni *vere* (m_i) sempre dalla formula 3.16, trascurando la condizione di esclusione $j \neq a$.

Si analizzi per ultimo caso un modello di Spin Glasses definito dall'Hamiltoniana seguente:

$$H = \sum_a \epsilon_a \prod_{j \in n(a)} s_j, \tag{3.17}$$

in cui $n(a)$ contiene l'insieme degli indici di spin in relazione con l'energia d'interazione ϵ_a.

Allo stesso modo con cui si è giunti a formulare l'equazione 3.16, si possono derivare le magnetizzazioni cave da:

$$m_{i;\sim a} = tanh\left[\sum_{k \in \partial i \setminus a} artanh(tanh(\beta \epsilon_k) \prod_{j \in n(k) \setminus i} m_{j;\sim k})\right]. \tag{3.18}$$

Dall'equazione 3.18 è possibile ricostruire in maniera diretta la relazione con le equazioni di Belief Propagation ([16]).

3.3.2 Relazione con la BP

I modelli di approssimazioni di campo medio, accennati in precedenza, si radicano su ipotesi che consentono di fattorizzare la distribuzione di probabilità degli spin. Ciò implica che è possibile rappresentare il sistema di spin attraverso i grafi fattoriali, sui quali lo stesso algoritmo di Belief Propagation è stato esposto nel capitolo 2.

Si valuti nuovamente il modello di Spin Glasses descritto dall'Hamiltoniana 3.17. Un sistema di questo genere può essere rappresentato genericamente dalla figura 3.1, nella quale i termini di interazione ϵ_a e gli spin sono indicati rispettivamente da $\square$ e da $\bigcirc$.

É possibile dedurre che i termini d'interazione ϵ_a rivestono il ruolo di nodi fattore mentre gli spin s_i di nodi variabile.

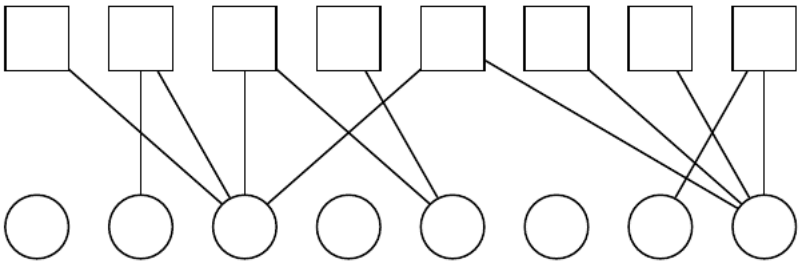

Figura 3.1: Esempio di grafo fattoriale per un modello di Spin Glasses.

L'equazione per ottenere le magnetizzazioni cave 3.18 può essere distinta in due quantità:

$$\begin{aligned} m_{i;\sim a} &= tanh\Big[\sum_{k\in\partial i\setminus a} artanh(\hat{m}_{k;\sim i})\Big] \\ \hat{m}_{a;\sim i} &= tanh(\beta\epsilon_a)\prod_{j\in n(a)\setminus i} m_{j;\sim a}. \end{aligned} \tag{3.19}$$

Si può osservare che $m_{i;\sim a}$ rappresenta la magnetizzazione cava i, ottenuta trascurando il termine d'interazione a; mentre $\hat{m}_{a;\sim i}$ rappresenta una stima del termine d'interazione a (che consente l'interazione tra gli spin indicizzati da $n(a)$), nel caso in cui si trascuri la dipendenza dallo spin i-esimo.

Le due quantità dipendono reciprocamente e ciò implica che, ad esempio, per calcolare $m_{i;\sim a}$ è necessario saper calcolare $\hat{m}_{k;\sim i}$; il quale dipende dai termini $m_{j;\sim k}$ nella produttoria dell'equazione 3.18. É da questi ultimi termini che si ritrova $m_{i;\sim a}$ e per questo devono già essere stati determinati.

Per descrivere il fatto che alcune quantità devono essere già conosciute, si definisce una sorta di tempo t e si riportano le equazioni in 3.19 nella forma:

$$\begin{aligned} m^t_{i;\sim a} &= tanh\Big[\sum_{k\in\partial i\setminus a} artanh(\hat{m}^t_{k;\sim i})\Big] \\ \hat{m}^{t+1}_{a;\sim i} &= tanh(\beta\epsilon_a)\prod_{j\in n(a)\setminus i} m^t_{j;\sim a}. \end{aligned} \tag{3.20}$$

Ora, si riprendano in considerazione le equazioni di aggiornamento della BP 2.8 e 2.9,

che si riportano per comodità:

$$\begin{aligned} \nu^t_{x_i \to f_a}(x_i) &= \prod_{k \in \partial x_i \setminus a} \hat{\nu}^t_{f_k \to x_i}(x_i) \\ \hat{\nu}^{t+1}_{f_a \to x_i}(x_i) &= \sum_{\sim x_i} [f_a(\vec{X}_a) \prod_{j \in n(f_a) \setminus i} \nu^t_{x_j \to f_a}(x_j)]. \end{aligned} \tag{3.21}$$

Come spiegato nella sezione 2.2.2, l'algoritmo di BP consiste tipicamente nell'aggiornamento in alternanza delle equazioni 3.21 fino ad un certo numero di step oppure fino a convergenza, con l'obiettivo di stimare distribuzioni di probabilità marginali. Inoltre, nella sezione 2.3.2 è stato identificato il significato dei messaggi, ovvero in cosa effettivamente consistono gli oggetti scambiati fra i nodi.

Prima di individuare la corrispondenza, è necessario adattare il sistema su cui è stata introdotta la Belief Propagation (capitolo 2) al sistema di Spin Glasses.

Il problema di ricavare le magnetizzazioni locali, visto in precedenza, è equivalente ad un problema di inferenza Bayesiana in cui la distribuzione di probabilità sugli spin è condizionata dalla conoscenza dei parametri che governano le interazioni tra di essi.

Si consideri quindi l'Hamiltoniana in equazione 3.17, si fissi una distribuzione di probabilità a priori $P(\vec{s})$ uniforme e si noti che i parametri d'interazione sono $\{\epsilon_a\}$. Supponendo che tali parametri siano estratti indipendentemente da distribuzioni $P(\epsilon_a)$, la distribuzione di probabilità studiata è :

$$P(\vec{s}|\{\epsilon_a\}) \propto P(\{\epsilon_a\}|\vec{s}) = \prod_a P(\epsilon_a|\vec{s}). \tag{3.22}$$

In questa formula non è inserito il termine di normalizzazione e si definiscono le funzioni $P(\epsilon_a|\vec{s})$ come:

$$P(\epsilon_a|\vec{s}) = \frac{e^{\beta\epsilon_a \prod_{j \in n(\epsilon_a)} s_j}}{\sum_{\epsilon_a} e^{\beta\epsilon_a \prod_{j \in n(\epsilon_a)} s_j}}. \tag{3.23}$$

Come si può notare dalla formula 3.22, la distribuzione $P(\vec{s}|\{\epsilon_a\})$ è fattorizzabile in funzioni, analogamente alla forma di $P(\vec{s})$ in 2.16. In questa situazione le variabili x_i sono gli spin s_i; mentre le funzioni $f_a(\vec{X}_a)$ in 2.16 sono proprio le probabilità condizionate $P(\epsilon_a|\vec{s})$.

Immaginando di rappresentare il sistema in un grafo fattoriale, oltre ad inserire i nodi variabili s_i, si è soliti indicare per semplicità i nodi fattori direttamente con i parametri ϵ_a da cui dipendono le funzioni $P(\epsilon_a|\vec{s})$. Sotto queste condizioni i messaggi in 3.21 divengono:

$$\begin{aligned} \nu^t_{s_i \to \epsilon_a}(s_i) &= \prod_{k \in \partial s_i \setminus a} \hat{\nu}^t_{\epsilon_k \to s_i}(s_i) \\ \hat{\nu}^{t+1}_{\epsilon_a \to s_i}(s_i) &= \sum_{\sim s_i} [P(\epsilon_a|\vec{s}) \prod_{j \in n(\epsilon_a) \setminus i} \nu^t_{s_j \to \epsilon_a}(s_j)]. \end{aligned} \tag{3.24}$$

Approfondendo per questo problema di inferenza Bayesiana il significato dei messaggi (trattato nella sezione 2.3.2), si possono notare le seguenti relazioni probabilistiche [17–19]:

$$\begin{aligned} P(s_i|\{\epsilon_{k\neq a}\}) &= \nu^t_{s_i\to\epsilon_a}(s_i) \\ P(\epsilon_a|s_i,\{\epsilon_{k\neq a}\}) &= \sum_{\sim s_i} P(\epsilon_a|\vec{s})P(\vec{s}|\{\epsilon_{k\neq a}\}) \propto \hat{\nu}^t_{\epsilon_a\to s_i}(s_i). \end{aligned} \tag{3.25}$$

Ora si parametrizzino i messaggi $\nu^t_{s_i\to\epsilon_a}$ e $\hat{\nu}^t_{\epsilon_a\to s_i}$ con le seguenti quantità $m^t_{s_i\to\epsilon_a}$ e $\hat{m}^t_{\epsilon_a\to s_i}$:

$$\begin{aligned} m^t_{s_i\to\epsilon_a} &= \frac{\sum_{s_i} s_i \nu^t_{s_i\to\epsilon_a}(s_i)}{\sum_{s_i}\nu^t_{s_i\to\epsilon_a}(s_i)} \\ \hat{m}^t_{\epsilon_a\to s_i} &= \frac{\sum_{s_i} s_i \hat{\nu}^t_{\epsilon_a\to s_i}(s_i)}{\sum_{s_i}\hat{\nu}^t_{\epsilon_a\to s_i}(s_i)}, \end{aligned} \tag{3.26}$$

e, dato che le variabili $\{s_i\}$ sono binarie, tali quantità possono riscritte come:

$$\begin{aligned} m^t_{s_i\to\epsilon_a} &= \frac{\nu^t_{s_i\to\epsilon_a}(+1) - \nu^t_{s_i\to\epsilon_a}(-1)}{\nu^t_{s_i\to\epsilon_a}(+1) + \nu^t_{s_i\to\epsilon_a}(-1)} \\ \hat{m}^t_{\epsilon_a\to s_i} &= \frac{\hat{\nu}^t_{\epsilon_a\to s_i}(+1) - \hat{\nu}^t_{\epsilon_a\to s_i}(-1)}{\hat{\nu}^t_{\epsilon_a\to s_i}(+1) + \hat{\nu}^t_{\epsilon_a\to s_i}(-1)}. \end{aligned} \tag{3.27}$$

Si noti che la parametrizzazione eseguita richiama la definizione vista per le magnetizzazioni locali nella formula 3.6.

Ora, basandosi su 3.23 ed eseguendo alcuni passaggi algebrici (mostrati esplicitamente nelle appendici C e D), si può ottenere:

$$\begin{aligned} m^t_{s_i\to\epsilon_a} &= tanh\left[\sum_{k\in\partial s_i\setminus a} artanh(\hat{m}^t_{\epsilon_k\to s_i})\right] \\ \hat{m}^{t+1}_{\epsilon_a\to s_i} &= tanh(\beta\epsilon_a)\prod_{j\in n(\epsilon_a)\setminus i} m^t_{s_j\to\epsilon_a}. \end{aligned} \tag{3.28}$$

In generale, grazie alla parametrizzazione in 3.27, i messaggi si possono esprimere come[6] :

$$\begin{aligned} \nu^t_{s_i\to\epsilon_a}(s_i) &= \frac{1 + m^t_{s_i\to\epsilon_a}\, s_i}{2} \\ \hat{\nu}^t_{\epsilon_a\to s_i}(s_i) &\propto \frac{1 + \hat{m}^t_{\epsilon_a\to s_i}\, s_i}{2}. \end{aligned} \tag{3.29}$$

Considerando che le equazioni 3.28 sono formalmente uguali alle equazioni per le magnetizzazioni cave 3.20, si può concludere che esiste una corrispondenza tra gli Spin Glasses e l'algoritmo di Belief Propagation.

[6]Si noti che per $\hat{\nu}^t_{\epsilon_a\to s_i}(s_i)$, in equazione 3.29, si è utilizzato il simbolo $\propto$ per indicare la mancanza del termine di normalizzazione rispetto ai valori assumibili da ϵ_a.

3.3.3 Caso generale

Nel caso in cui il sistema studiato non sia governato dallo stesso tipo di funzioni 3.23, la parametrizzazione delle equazioni di BP attraverso le magnetizzazioni può essere comunque eseguita. In particolare, l'equazione relativa ai messaggi da nodo variabile a nodo fattore $m^t_{s_i \to \epsilon_a}$ rimane nella stessa forma in 3.28; mentre diventa più elaborata l'equazione per $\hat{m}^{t+1}_{\epsilon_a \to s_i}$.

In conclusione, le equazioni di Belief Propagation, parametrizzate attraverso le magnetizzazioni cave (e considerando variabili di Ising), hanno forma generale:

$$\begin{aligned} m^t_{x_i \to f_a} &= tanh\left[\sum_{k \in \partial x_i \setminus a} artanh(\hat{m}^t_{f_k \to x_i}) \right] \\ \hat{m}^{t+1}_{f_a \to x_i} &= \frac{\sum_{\vec{X}_a} \left[x_i \, f_a(\vec{X}_a) \prod_{j \in n(f_a) \setminus i} \left(\frac{1 + m^t_{x_j \to f_a}}{2} \right) \right]}{\sum_{\vec{X}_a} \left[f_a(\vec{X}_a) \prod_{j \in n(f_a) \setminus i} \left(\frac{1 + m^t_{x_j \to f_a}}{2} \right) \right]}. \end{aligned} \tag{3.30}$$

Capitolo 4

Studio dell'apprendimento: dall'equilibrio al non-equilibrio

Questo capitolo riassume e spiega i risultati ottenuti da quel settore di ricerca in Fisica che mira a modellizzare il fenomeno dell'apprendimento attraverso un modello fisico-statistico. Le teorie alla base di questo settore provengono principalmente dalla Meccanica Statistica e dallo studio dei sistemi di Spin Glasses.

Nella prima sezione è spiegato e categorizzato il fenomeno dell'apprendimento ed è introdotto il sistema mediante cui si modellizza tale fenomeno.

La seconda sezione evidenzia la necessità di considerare effetti al non-equilibrio delineando la difficoltà che emerge all'equilibrio.

Nella terza sezione sono infine esposti i risultati degli studi di Large Deviation Analysis svolti per descrivere gli effetti al non-equilibrio.

4.1 Approccio all'apprendimento

Il fenomeno dell'apprendimento può essere classificato in due tipologie: un'apprendimento atto ad immagazzinare informazioni qualsiasi ed un'apprendimento atto ad imparare processi. Innanzitutto, si considerino per semplicità M vettori $\{\vec{\xi}^{\mu}\}_{\mu=1}^{M}$, ognuno di dimensione pari a N, e si assegni a ciascun di tali vettori un'etichetta τ^{μ}. Si identifichi inoltre l'associazione generica $(\vec{\xi}, \tau^{\mu})$ con il termine *pattern*.

Il primo tipo di apprendimento può essere interpretato come una sorta di memorarizzazione completa (*storage*) di un set di patterns random, dove il termine random indica che i patterns sono estratti da una distribuzione di probabilità. Il secondo tipo di apprendimento consiste invece nel riconoscere dalla natura dei patterns un processo o una regola, che possa poi eventualmente essere generalizzata su altri patterns. Solitamente a queste due tipologie di apprendimento corrispondono rispettivamente il *problema di classificazione* e il *problema di generalizzazione*.

L'approccio alla risoluzione di questi problemi sfrutta come apparato matematico di base le Neural Networks feed-forward fully-connected [20], introdotte nel capitolo 1. Le NNs feed-forward fully-connected trattate, indicate nel resto del testo semplicemente con NNs, sono strumenti di calcolo che ricevono un vettore in input, ad esempio $\vec{\xi^{\mu}}$, trasformano le componenti di tale vettore attraverso funzioni non lineari e producono una quantità scalare in output.

A questo punto, è possibile formalizzare un sistema su cui modellizzare i tipi di apprendimento. Le variabili $\vec{\sigma}$ che descrivono il sistema sono le variabili che governano il funzionamento delle NNs, cioè i pesi sinaptici. L'energia del sistema si definisca invece genericamente con la formula:

$$E(\vec{\sigma}; \{\vec{\xi}\}, \{\tau^{\mu}\}) = \sum_{\mu=1}^{M} E^{\mu}(\vec{\sigma}) \ , \tag{4.1}$$

in cui $E^{\mu}(\vec{\sigma}; \{\vec{\xi}\}, \{\tau^{\mu}\})$=0 se la NNs produce l'output corretto per il vettore $\vec{\xi^{\mu}}$ e $E^{\mu}(\vec{\sigma}; \{\vec{\xi}\}, \{\tau^{\mu}\})$=1 altrimenti. In altre parole, l'energia del sistema è definita come il numero di patterns sbagliati da una configurazione di pesi $\vec{\sigma}$.

La convenzione adottata per l'equazione 4.2 è la stessa del capitolo precedente. Sono anteposte e posposte al ";" rispettivamente le variabile e i parametri del sistema. Anche in questo capitolo, per alleggerire la trattazione in futuro sono riportati come argomenti delle funzioni solo le variabili, trascurando i parametri. Perciò $E(\vec{\sigma}; \{\vec{\xi}\}, \{\tau^{\mu}\})$ è riportata come $E(\vec{\sigma})$.

4.1.1 Problema di classificazione

Data una distribuzione di probabilità $P(\vec{\xi^{\mu}}, \tau^{\mu})$ da cui estrarre indipendentemente M patterns, il problema di classificazione si impone di trovare un set di pesi $\vec{\sigma}$ tali per cui tutti gli M patterns siano memorizzati. Nell'ottica del sistema fisico, ciò significa che il problema è risolto con configurazioni $\vec{\sigma}$ tali per cui $E(\vec{\sigma})$=0.

Ora, per semplicità, si considerino tutte le grandezze coinvolte nell'apprendimento come variabili di Ising (± 1). A questo punto, si può definire:

$$E^{\mu}(\vec{\sigma}) = \Theta(-\tau^{\mu}\, \tau(\vec{\sigma}, \vec{\xi^{\mu}})) \ , \tag{4.2}$$

in cui τ^{μ} è il valore in output atteso e $\tau(\vec{\sigma}, \vec{\xi^{\mu}})$ è il valore in output calcolato dalla NN. La funzione $\Theta(x)$ è invece la funzione di Heaviside:

$$\begin{cases} 0 \ , per \ x < 0 \\ 1 \ , per \ x \geq 0 \ . \end{cases}$$

Pertanto, si ha $\Theta(-\tau^\mu\,\tau(\vec{\sigma},\vec{\xi^\mu}))=0$ se $\tau^\mu=\tau(\vec{\sigma},\vec{\xi^\mu})$ e $\Theta(-\tau^\mu\,\tau(\vec{\sigma},\vec{\xi^\mu}))=1$ altrimenti. É inoltre utile definire la quantità:

$$\chi(\vec{\sigma}) = \prod_{\mu=1}^{M} \Theta(\tau^\mu\,\tau(\vec{\sigma},\vec{\xi^\mu})), \tag{4.3}$$

che è pari a 1 solo se tutti i patterns sono classificati correttamente da $\vec{\sigma}$ ed è 0 altrimenti. Tramite la quantità $\chi(\vec{\sigma})$ la risoluzione del problema di classificazione si può interpretare equivalentemente anche come la determinazione di configurazioni $\vec{\sigma}$ per cui $\chi(\vec{\sigma}) = 1$.

Classificazione su percettrone

Considerando sempre variabili di Ising, si supponga di voler risolvere il problema di classificazione su un percettrone; il cui valore in output è calcolato da:

$$\tau(\vec{\sigma},\vec{\xi^\mu})) = sgn(\sum_{i=1}^{N} \sigma_i\,\xi_i^\mu). \tag{4.4}$$

Si riscrivano l'energia espressa in 4.1 come:

$$E(\vec{\sigma}) = \sum_{\mu=1}^{M} \Theta\Big(- \tau^\mu \sum_{i=1}^{N} \sigma_i\,\xi_i^\mu\Big)\ , \tag{4.5}$$

e:

$$\chi(\vec{\sigma}) = \prod_{\mu=1}^{M} \Theta\Big(\tau^\mu \sum_{i=1}^{N} \sigma_i\,\xi_i^\mu\Big). \tag{4.6}$$

Ora, si definisca il *volume di Gardner* [20, 21] come:

$$\Omega(\{\vec{\xi^\mu}\},\{\tau^\mu\}) = \sum_{\{\vec{\sigma}\}} \chi(\vec{\sigma}) = \sum_{\{\vec{\sigma}\}} \prod_{\mu=1}^{M} \Theta\Big(\tau^\mu \sum_{i=1}^{N} \sigma_i\,\xi_i^\mu\Big). \tag{4.7}$$

Questo volume calcola il numero di soluzioni esistenti al problema di classificazione, una volta fissati i pattern. Siccome tali pattern sono quantità random, solitamente si analizza il volume di Gardner tramite una cosiddetta densità d'entropia:

$$S = \frac{1}{N}\langle ln\Omega(\{\vec{\xi^\mu}\},\{\tau^\mu\})\rangle_{\{\vec{\xi^\mu}\},\{\tau^\mu\}}\ , \tag{4.8}$$

riprendendo la notazione $\langle\rangle$ vista nel capitolo precedente.

Il calcolo di questa quantità può essere attuato attraverso il metodo delle repliche e può essere condotto sia ipotizzando la simmetria tra le repliche (RS) che ipotizzando una rottura della simmetria (RSB). Una quantità rilevante in questo calcolo è la *densità*

vincolata α, uguale al rapporto tra il numero di pattern e il numero di sinapsi (la dimensione dei vettori): $\alpha = \frac{M}{N}$. Si noti che questa variabile è chiamata anche *capacità* e che frequentemente il limite superiore di α è definito *capacità massima* (α_C).

É stato dimostrato [22] grazie all'ipotesi di RS, che esistono configurazioni del sistema con $S < 0$ per $\alpha > 0.83$ e che pure utilizzando il primo step di RSB (1-RSB) il valore critico di α è ~ 0.83. Di conseguenza, la capacità massima di un percettrone binario è ritenuta fino ad oggi $\alpha_C \simeq 0.83$. In altri termini, il problema di classificazione di M pattern random può essere sempre risolto per $\alpha \leq \alpha_C$ e non può mai essere risolto $\alpha > \alpha_C$.

Questi passaggi sono stati mostrati su un percettrone binario per semplicità, ma generalmente si possono ottenere risultati analoghi per NNs con strutture più complesse e con quantità discrete con più stati.

4.1.2 Problema di generalizzazione

Data una NN con pesi sinaptici fissati, chiamata NN *teacher*, il problema di generalizzazione è risolto identificando il funzionamento di tale NN. In pratica, il problema di generalizzazione consiste nel considerare una NN *student* con pesi $\vec{\sigma}^S$ e una NN *teacher* con pesi $\vec{\sigma}^T$, dotate della stessa struttura, e nell'allenare la prima in modo che emuli la seconda [1] .

Il training set è ricavato estraendo M vettori $\vec{\xi}^\mu$ indipendentemente da una distribuzione $P(\vec{\xi}^\mu)$ e calcolando i relativi valori in output usando la NN teacher. Siccome l'obiettivo è ottenere una NN student che emuli la NN teacher, l'obiettivo della fase di training è allenare la NN student in modo che calcoli in output gli stessi valori in output della NN teacher. Nella prospettiva del sistema fisico, così come per il problema di classificazione, ciò significa trovare $\vec{\sigma}^S$ tale per cui l'energia è nulla. Si noti però, che il fatto di trovare una di tali configurazioni con energia nulla non implica necessariamente che la NN student funzioni esattamente come la NN teacher. In altre parole, esiste uno spazio di NNs student compatibili con la NN teacher sui pattern del training set. Matematicamente, questo spazio è definito da: $\{\vec{\sigma}^S \neq \vec{\sigma}^T \mid E(\vec{\sigma}^S) = 0\}$.

Anche per questo problema, si considerino tutte le grandezze coinvolte nell'apprendimento come variabili di Ising (± 1). L'argomento della sommatoria in equazione 4.1, diventa per il problema di generalizzazione pari a:

$$E^\mu(\vec{\sigma}^S) = \Theta(-\tau^T(\vec{\sigma}^T, \vec{\xi}^\mu)\, \tau^S(\vec{\sigma}^S, \vec{\xi}^\mu)) \ , \tag{4.9}$$

in cui le funzioni $\tau^T(\vec{\sigma}^T, \vec{\xi}^\mu)$ e $\tau^S(\vec{\sigma}^S, \vec{\xi}^\mu)$ indicano i valori calcolati rispettivamente dalla NN teacher e dalla NN student.

[1] Questo tipo di studio si può anche estendere a casi dove la struttura e le proprietà tra la NN student e la NN teacher non sono le stesse. Per semplicità in questo testo è riportato solo il caso di uguaglianza tra le NNs.

Ora, si consideri una NN student con configurazione $\vec{\sigma}^S$ compatibile con $\vec{\sigma}^T$. Il fatto che la NN student riesca a risolvere perfettamente il training set nonostante $\vec{\sigma}^S \neq \vec{\sigma}^T$, suggerisce l'esistenza di una sorta di corrispondenza tra la NN student e la NN teacher. Questa corrispondenza tra i pesi $\vec{\sigma}^S$ della NN student e quelli della NN teacher $\vec{\sigma}^S$ è formalizzata dal cosiddetto *overlap*, definito come:

$$q = \frac{1}{N}(\vec{\sigma}^S \cdot \vec{\sigma}^T) \ . \tag{4.10}$$

É possibile dimostrare che l'overlap consente di determinare la probabilità che la NN student commetta un errore rispetto alla NN teacher su un pattern non appartenente al training set [23]. Questa probabilità, indicata solitamente con l'espressione *errore di generalizzazione*, si può calcolare grazie alla formula:

$$p_e(q) = \frac{1}{\pi} arccos(q) \ . \tag{4.11}$$

Quindi, la risoluzione del problema di generalizzazione aspira a trovare, una volta concluso il training, una configurazione di pesi compatibile avente un basso errore di generalizzazione.

Generalizzazione su percettrone

Si considerino, sempre per semplicità, variabili di Ising e si supponga di voler risolvere il problema di generalizzazione su un percettrone. Si riscriva l'energia espressa in 4.1 come:

$$E(\vec{\sigma}^S) = \sum_{\mu=1}^{M} \Theta\Big(- (\vec{\sigma}^T \cdot \vec{\xi}^\mu)\,(\vec{\sigma}^S \cdot \vec{\xi}^\mu)\Big) \ . \tag{4.12}$$

Anche in questo problema è possibili identificare la quantità:

$$\chi(\vec{\sigma}^S) = \prod_{\mu=1}^{M} \Theta\Big(- (\vec{\sigma}^T \cdot \vec{\xi}^\mu)\,(\vec{\sigma}^S \cdot \vec{\xi}^\mu)\Big). \tag{4.13}$$

Si definisca quindi il *volume di Gardner* del problema come:

$$\Omega(\{\vec{\xi}^\mu\}, \vec{\sigma}^T) = \sum_{\{\vec{\sigma}^S\}} \chi(\vec{\sigma}^S) = \sum_{\{\vec{\sigma}^S\}} \prod_{\mu=1}^{M} \Theta\Big(- (\vec{\sigma}^T \cdot \vec{\xi}^\mu)\,(\vec{\sigma}^S \cdot \vec{\xi}^\mu)\Big). \tag{4.14}$$

Come per il problema di classificazione, questo volume calcola il numero di soluzioni esistenti al problema di generalizzazione, una volta determinati i vettori di input random e i pesi $\vec{\sigma}^T$ del percettrone teacher. In base a quanto visto in precedenza, l'analisi del volume di Gardner è effettuata dal calcolo della densità d'entropia:

$$S = \frac{1}{N} \langle ln\Omega(\{\vec{\xi}^\mu\}, \vec{\sigma}^T) \rangle_{\{\vec{\xi}^\mu\}, \vec{\sigma}^T} \ . \tag{4.15}$$

In questo caso però la media $\langle\rangle_{\{\vec{\xi}^\mu\},\vec{\sigma}^T}$ è effettuata, oltre che sui vettori $\vec{\xi}^\mu$, anche sui pesi $\vec{\sigma}^T$ della NN teacher assumendo una distribuzione di probabilità $P(\vec{\sigma}^T)$.

La capacità $\alpha = \frac{M}{N}$ risulta sempre la quantità più interessante su cui studiare l'andamento dell'entropia S. Nonostante ciò, siccome l'interesse maggiore del problema di generalizzazione è individuare configurazioni $\vec{\sigma}^S$ compatibili con un basso errore di generalizzazione, è possibile studiare anche l'andamento medio dell'errore di generalizzazione al variare di α.

Sia da ipotesi di RS che di RSB, è possibile ritrovare che $S > 0$ per $\alpha < 1.245$ e che per $\alpha \geq 1.245$, S=0. Ciò significa che per $\alpha < 1.245$ esistono dei percettroni student compatibili con il percettrone teacher, mentre per $\alpha \geq 1.245$ non esistono tali altri percettroni ma solo il percettrone teacher. Allo stesso tempo l'errore di generalizzazione decresce progressivamente nel range $0 \leq \alpha < 1.245$, per poi saltare in maniera discontinua a 0 nel punto $\alpha = 1.245$, chiamato α_{TS}, e rimanere costante a 0 per valori superiori. In altre parole, ciò significa che un percettrone student in grado di produrre per $M = 1.245N$ vettori in input esattamente gli stessi output del percettrone teacher, coincide esattamente con quest'ultimo: $\vec{\sigma}^S = \vec{\sigma}^T$.

Analogamente al problema di classificazione, questi passaggi sono stati mostrati su un percettrone binario per semplicità, ma si possono estendere anche a NNs con strutture più complesse e caratterizzate da quantità discrete non binarie o continue.

4.2 Evoluzione degli algoritmi d'apprendimento

La conoscenza delle soglie α_C e α_{TS} permette di valutare le prestazioni dell'implementazione di un algoritmo di risoluzione al crescere del parametro α sia per il problema di classificazione che per il problema di generalizzazione. Un algoritmo è tipicamente valutato stabilendo un limite massimo di operazioni che esso può svolgere e analizzando i risultati ottenuti una volta raggiunto tale limite. Infatti, il semplice raggiungimento di una soluzione non può essere l'unico criterio di valutazione di un algoritmo, altrimenti anche l'implementazione di una ricerca puntuale sulle 2^N possibili configurazioni è potenzialmente capace di cadere in una soluzione. Di conseguenza, è la Teoria della Complessità Computazionale (sezione 1.4) che fornisce gli strumenti opportuni per valutare le prestazioni di un algoritmo di risoluzione.

Usando pesi continui, è possibile dimostrare che gli algoritmi per la risoluzione del problema di classificazione e per il problema di generalizzazione sono in grado di risolvere il problema con complessità ottimale. Nel caso invece si utilizzino pesi discreti la situazione cambia. Mentre le soluzioni del problema di generalizzazione rimangono ottimamente raggiungibili, è noto che la risoluzione del problema di classificazione diventi un problema NP-completo anche nel caso del percettrone binario [8, 24]. In altre parole, il problema di classificazione diventa intrattabile se un algoritmo mira a minimizzare progressivamente l'energia in equazione 4.1 . Gli algoritmi di questo tipo non sono in grado

di memorizzare un numero di pattern maggiore di lnM per un taglio sub-esponenziale di operazioni.

Ciònonostante, esistono una serie di algoritmi euristici [25–29] che invece risolvono lo stesso problema senza mostrare la stessa difficoltà computazionale [2] . Tali algoritmi euristici sono spesso chiamati in letteratura *algoritmi euristici efficaci*.

Per spiegare il motivo per cui esistono questi algoritmi efficaci è stato necessario effettuare un confronto tra le soluzioni attese teoricamente e le soluzioni trovate dagli algoritmi euristici. Si noti che questo confronto è stato svolto solo sul percettrone binario, assumendo che risultati analoghi si possano ottenere con NNs più complicate.

4.2.1 Soluzioni all'equilibrio

Gli algoritmi basati sulla minimizzazione dell'energia raggiungono tipicamente configurazioni che minimizzano localmente l'energia, ovvero configurazioni che non riescono a memorizzare tutto l'insieme di M pattern random. Questa situazione è facilmente contestualizzabile in un sistema di Spin Glasses. Come spiegato nel capitolo 3, nello stato di Spin Glass gli spin interagiscono attraverso termini random che limitano il raggiungimento di uno stato di minimo globale.

Si riprenda l'Hamiltoniana di un sistema di Spin Glasses vista in equazione 3.9:

$$H(\vec{S};\{\epsilon_{ij}\}) \equiv H(\vec{S}) = -\sum_{(ij)\in B} \epsilon_{ij} s_i s_j, \tag{4.16}$$

e l'energia del percettrone binario in equazione 4.5:

$$E(\vec{\sigma}) = \sum_{\mu=1}^{M} \Theta\Big(-\tau^{\mu} \sum_{i=1}^{N} \sigma_i \, \xi_i^{\mu} \Big) \ . \tag{4.17}$$

Nel caso del sistema di Spin Glasses sono i termini d'interazione ϵ_{ij} ad essere estratti casualmente, mentre nel percettrone sono i pattern. In entrambi i casi però, per raggiungere lo stato di minimo globale è necessario che i sistemi si configurino in modo da rispettare tutti i vincoli. Per il sistema di Spin Glasses ciò significa che tutte le coppie di spin devono orientarsi nella direzione vincolata dai ϵ_{ij}. Invece, per il percettrone ciò significa che la configurazione di pesi deve soddisfare ogni pattern [3] .

Trattando quindi il problema di classificazione sul percettrone binario come un sistema di Spin Glasses, è possibile analizzare le proprietà all'equilibrio delle soluzioni. In

[2] Il primo algoritmo euristico creato, chiamato *reinforced Belief Propagation* (rBP) ha mostrato una complessità computazionale pari a $O(N^2\, ln(N))$ e una capacità massima $\alpha_C \simeq 0.7$.

[3] Si noti che tali problemi, in cui si richiede di trovare uno stato del sistema che soddisfi una serie di vincoli, sono chiamati *Constraint Satisfaction Problems* (CSPs), già citati alla fine del capitolo 1. Inoltre, al di sotto delle soglie teoriche fissate da α_C, esistono soluzioni a tale problema; che quindi si può intendere come un problema SAT.

Meccanica Statistica lo studio delle proprietà all'equilibrio di un sistema governato da effetti stocastici (sezione 1.1) è descritto da un ensemble canonico utilizzando la distribuzione di Boltzmann 3.3. In particolare, il tentativo di raggiungere uno stato di minimo globale dell'energia suggerisce lo studio del sistema nel limite di $\beta \to \infty$. In questo limite, infatti, il sistema è vincolato a raggiungere il minimo energetico.

Lo studio delle proprietà all'equilibrio delle soluzioni è solitamente svolto tramite l'analisi del cosiddetto *potenziale di Franz-Parisi* [30]. L'obiettivo di questo studio è analizzare le proprietà medie degli intorni delle soluzioni all'equilibrio. Gli intorni sono definiti dalla distanza tra due configurazioni:

$$d(\vec{\sigma^a}, \vec{\sigma^b}) = \frac{1}{2}\sum_{i=1}^{N}(\sigma_i^a - \sigma_i^b)^2. \tag{4.18}$$

In particolare, nel caso binario è usata la *distanza di Hamming*:

$$d(\vec{\sigma^a}, \vec{\sigma^b}) \equiv d(q) = \frac{1-q}{2}, \tag{4.19}$$

che è proporzionale alla distanza espressa in equazione 4.18 e in cui q esprime l'overlap tra le due configurazioni.

Ora, data una distanza D, si definisca la cosiddetta *densità di entropia locale* per una configurazione generica di riferimento $\vec{\sigma}^\star$ come:

$$S(\vec{\sigma}^\star) = \frac{1}{N} ln\left(\sum_{\vec{\sigma}} e^{-\beta E(\vec{\sigma})}\, \delta\Big(d(\vec{\sigma}, \vec{\sigma^\star}), D\Big)\right), \tag{4.20}$$

in cui δ è la delta di Kroenecker e l'energia $E(\vec{\sigma})$ è data dall'equazione 4.17.

L'argomento del logaritmo può essere inteso come la funzione di partizione di un sistema all'equilibrio, le cui configurazioni sono vincolate ad una distanza D dalla configurazione di riferimento $\vec{\sigma}^\star$. Per ottenere un valore tipico per la densità di entropia locale è necessario considerare il fatto che le configurazioni di riferimento $\vec{\sigma}^\star$ e i pattern sono variabili random. Siccome il sistema è studiato all'equilibrio si assuma che le configurazioni di riferimento siano state estratte da una generica distribuzione di Boltzmann con β'. Sviluppando il termine dell'energia all'esponente la distribuzione dei $\vec{\sigma}^\star$ diventa:

$$P(\vec{\sigma}^\star) = \frac{1}{Z}\prod_{\mu=1}^{M} e^{-\beta'\Theta(-\tau^\mu\, \tau(\vec{\sigma}^\star, \vec{\xi}^\mu))}. \tag{4.21}$$

A questo punto, il valore tipico per la densità di entropia locale può essere calcolata da:

$$S \equiv S(D, N, \beta, \beta') = \left\langle \sum_{\{\vec{\sigma}^\star\}} P(\vec{\sigma}^\star)\, S(\vec{\sigma}^\star) \right\rangle_{\{\xi^\mu\},\{\tau^\mu\}}, \tag{4.22}$$

dove $\langle\rangle_{\{\xi^\mu\},\{\tau^\mu\}}$ indicano lo svolgimento di una media rispetto ad un'ipotetica distribuzione di pattern $P(\vec{\xi^\mu}, \tau^\mu)$.

Visto che il problema di classificazione si focalizza sullo stato di minimo globale per l'energia, si considerino i limiti $\beta \to \infty$ e $\beta' \to \infty$. Utilizzando l'identità

$$lim_{\beta\to\infty} e^{-\beta\Theta(-u)} = \Theta(u) \ ,$$

le equazioni 4.20, 4.21 e 4.22 diventano rispettivamente:

$$S(\vec{\sigma}^\star) = \frac{1}{N} ln\left(\sum_{\vec{\sigma}} \chi(\vec{\sigma})\, \delta\Big(d(\vec{\sigma}, \vec{\sigma^\star}), D\Big)\right) \ , \tag{4.23}$$

$$P(\vec{\sigma}^\star) = \frac{\chi(\vec{\sigma}^\star)}{Z} \ , \tag{4.24}$$

$$S \equiv S(D, N) = \left\langle \sum_{\{\vec{\sigma}^\star\}} P(\vec{\sigma}^\star)\, S(\vec{\sigma}^\star) \right\rangle_{\{\xi^\mu\},\{\tau^\mu\}} \ . \tag{4.25}$$

La quantità $S(\vec{\sigma}^\star)$ calcola fondamentalmente il logaritmo naturale, normalizzato da N, del numero di soluzioni ad una distanza normalizzata di Hamming D (equazione 4.19) attorno alla soluzione di riferimento $\vec{\sigma}^\star$. Si ricordi infatti che $\chi(\vec{\sigma}^\star)$ e $\chi(\vec{\sigma})$ sono diversi da 0 solo se i pesi all'argomento sono soluzioni del problema. La grandezza S rappresenta invece il valore tipico di questo numero una volta mediato sia sui pattern che sui pesi.

La risoluzione analitica dell'equazione 4.23 può essere condotta analogamente a quanto operato per il calcolo del volume di Gardner [31]. Dopo aver utilizzato il metodo delle repliche e aver assunto un'ipotesi di RS o RSB, è possibile ricavare una forma approssimativa per S. Questa quantità esprime il potenziale di Franz-Parisi, ovvero l'andamento tipico del numero di soluzioni attorno ad una soluzione di riferimento.

É possibile anche, per semplicità, sostituire alla funzione 4.23 il delta di Kroenecker con una funzione vincolo meno rigida:

$$S(\vec{\sigma}^\star) = \frac{1}{N} ln\left(\sum_{\vec{\sigma}} \chi(\vec{\sigma})\, e^{\gamma \vec{\sigma}\cdot\vec{\sigma}^\star}\right) \ . \tag{4.26}$$

A questo punto, per ricondursi al potenziale di Franz-Parisi è necessario ricavare $S \equiv S(\gamma, N)$ come in equazione 4.25, risolvere il limite:

$$s(\gamma) = lim_{N\to\infty} S(\gamma, N) \tag{4.27}$$

e sfruttare la seguente trasformata di Legendre:

$$V(q) = s(\gamma) - \gamma\, q \tag{4.28}$$

in cui q è la soglia di overlap e determina l'intorno analogamente a D. In particolare si può dimostrare: $q = \frac{d\,s(\gamma)}{d\,\gamma}$. Ricordando che la distanza D di soglia si può ritrovare dalla distanza di Hamming normalizzata $D = \frac{1-q}{2N}$, il potenziale di Franz-Parisi si ritrova da $e^{NV(q)}$.

Dall'analisi del potenziale di Franz-Parisi al variare della distanza di soglia D si è dimostrato che le soluzioni all'equilibrio sono soluzioni isolate. In pratica, ciò significa che esiste una distanza minima D_{min} al di sotto della quale l'intorno tipico di una soluzione all'equilibrio è vuoto. Inoltre, è possibile osservare che D_{min} cresce all'aumentare di α, ovvero che vi è un ulteriore allontamento progressivo delle soluzioni al crescere della capacità (figura 4.1).

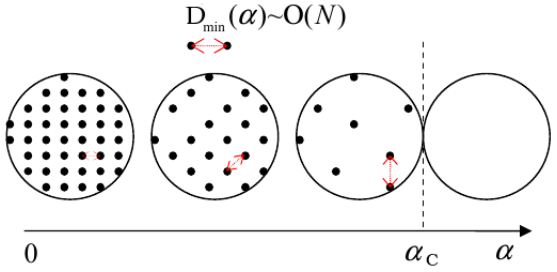

Figura 4.1: Evoluzione dello spazio delle soluzioni all'aumentare della capacità. Le soluzioni si allontanano mano a mano che α cresce fino a quando non è superata la capacità massima α_C, oltre la quale non esiste alcuna soluzione.

Come sottolineato dalla figura 4.1, la distanza minima varia proporzionalmente rispetto a N e quindi per passare da una soluzione all'altra è necessario effettuare N modifiche alle componenti.

4.2.2 Soluzioni algoritmiche

L'analisi delle soluzione degli algoritmi efficaci consiste invece nell'eseguire una *random walk* a partire da queste soluzioni e andare ad analizzare lo spazio attorno ad esse. Una random walk è un processo stocastico che traccia un percorso di punti secondo steps random. In questo caso, scelta una soluzione algoritmica $\vec{\sigma}^{\star}$, il singolo step ha consistito nel muoversi in un'altra configurazione modificando una singola componente random della soluzione (*single random spin flip*). Il processo procede secondo questo tipo di step fino a che la configurazione d'arrivo non è una soluzione.

É possibile dimostrare che si possono trovare un numero di soluzioni attorno ad una soluzione algoritmica fino a N random spin flips e che l'andamento di tale numero cresce esponenzialmente rispetto ad N. Ora, la densità di entropia locale di un sistema di soluzioni ad una certa distanza dalla soluzione algoritmica $\vec{\sigma}^{\star}$ (equazione 4.26) può essere stimata numericamente. Il risultato di questa stima è che anche per distanze

prossime allo 0 tale densità di entropia locale non si annulla. In letteratura, le soluzioni algoritmiche si dicono appartenenti a cluster densi di soluzioni.

4.2.3 Effetti al non-equilibrio

L'apparente contraddizione dovuta alla difficoltà computazionale degli algoritmi basati sulla minimizzazione dell'energia e alla contemporanea esistenza di algoritmi euristici efficaci, si può spiegare ammettendo che gli algoritmi euristici non seguano un andamento all'equilibrio. La differenza tra le soluzioni all'equilibrio e le soluzioni algoritmiche è che le prime sono attese isolate e le seconda appartengono a cluster densi di soluzioni.

Un'incogruenza così netta suggerisce che gli algoritmi euristici siano attirati da configurazioni caratterizzate da intorni ricchi di soluzioni e che questo tipo di attrazione renda più accessibili le soluzioni del problema di classificazione. Questo effetto non si può ottenere invece per gli algoritmi disegnati sulla minimizzazione diretta dell'energia.

La minimizzazione dell'energia del sistema è inevitabilmente un processo all'equilibrio e ciò significa utilizzare una distribuzione (Boltzmann) che pesa equamente tutte le configurazioni (sezione 4.2.1). La difficoltà computazionale mostrata dagli algoritmi tipici può essere quindi giustificata dal fatto che tale distribuzione pesa equamente tutte le configurazioni $\vec{\sigma}$ e che questa equalità proietti il sistema in uno spazio piatto, dove le soluzioni sono isolate tra loro. In altre parole, all'equilibrio ogni configurazione ha la stessa probabilità di realizzarsi. Si può ipotizzare quindi che l'efficienza degli algoritmi euristici è dovuta ad un effetto di non-equilibrio che rompe tale equalità.

Queste osservazioni suggeriscono che il meccanismo necessario per recuperare l'efficacia degli algoritmi efficaci è ripesare le configurazioni del sistema favorendo coloro che si ritrovano in regioni ricche di soluzioni. Questo tipo di impostazione entra fa parte dei cosiddetti studi di *Large Deviation Analysis* e può essere utilizzato come base dello sviluppo di nuovi algoritmi d'apprendimento.

4.3 Large Deviation Analysis

Gli studi di Large Deviation Analysis fanno capo alla cosiddetta *Large Deviation Theory*, che è il ramo della Meccanica Statistica che analizza fenomeni o processi rari e particolari. Come si evince dalla sezione precedente, gli algoritmi euristici efficaci identificano soluzioni con proprietà nettamente distinte da quelle attese da un descrizione all'equilibrio del sistema. Questi stati estranei all'equilibro suggeriscono l'uso di distribuzioni al non-equilibrio (diverse da quelle di Boltzmann) definite in base alle proprietà delle soluzione algoritmiche osservate. Siccome in Meccanica Statistica gli stati estranei o "invisibili" all'equilibrio sono spesso chiamati *stati sub-dominanti*, anche le soluzioni degli algoritmi euristici possono essere ritenute analogamente sub-dominanti.

La proprietà principale di tali soluzioni sub-dominanti è l'appartenenza a regioni ricche di altre soluzioni sub-dominanti. Fisicamente ciò significa che le soluzioni sub-dominanti sono caratterizzate da un alto valore per la densità di entropia locale $S(\vec{\sigma}^\star)$ (equazione 4.23).

In letteratura sono stati definiti due nuovi tipi di distribuzioni, focalizzati sulla densità di entropia locale, che hanno obiettivi ben distinti. La prima distribuzione P_{RC}, detta *distribuzione ripesata e vincolata*, è stata formulata per favorire il confronto analitico-teorico con la distribuzione di Boltzmann. La seconda distribuzione P_{RU}, detta *distribuzione ripesata e non-vincolata*, è stata invece creata come base di sviluppo per nuovi algoritmi di apprendimento. In letteratura entrambe le distribuzioni sono definite considerando quantità binarie.

4.3.1 Distribuzione ripesata e vincolata

La distribuzione di probabilità *ripesata* e *vincolata* P_{RC} è definita dalla formula:

$$P_{RC}(\vec{\sigma}^\star) = \frac{\chi(\vec{\sigma}^\star)\, e^{yNS(\vec{\sigma}^\star)}}{Z} \ . \tag{4.29}$$

Questa distribuzione è simile alla distribuzione di Boltzmann in equazione 4.24 ed è caratterizzata dal fattore $e^{yNS(\vec{\sigma}^\star)}$. Ora, si definisca la densità di energia libera vincolata tipica Φ_{RC} [4] :

$$\Phi_{RC} \equiv \Phi_{RC}(D, y, N, \beta) = \frac{1}{N} \left\langle ln \sum_{\vec{\sigma^\star}} \chi(\vec{\sigma^\star})\, e^{yNS(\vec{\sigma}^\star)} \right\rangle_{\{\xi^\mu\},\{\tau^\mu\}} . \tag{4.30}$$

In tale distribuzione, $\vec{\sigma}^\star$ agisce sempre da configurazione di riferimento, mentre y ha formalmente il ruolo di un inverso della temperatura. La densità di entropia locale per questo sistema, analoga all'equazione 4.25, è:

$$S_{RC} \equiv S_{RC}(D, y, N, \beta) = \left\langle \sum_{\{\vec{\sigma}^\star\}} P_{RC}(\vec{\sigma}^\star) S(\vec{\sigma}^\star) \right\rangle_{\{\xi^\mu\},\{\tau^\mu\}} , \tag{4.31}$$

che è legata a Φ_{RC} da:

$$S_{RC}(D, y, N, \beta) = \frac{\partial}{\partial y} \Phi_{RC}(D, y, N, \beta) \ . \tag{4.32}$$

L'equazione 4.32 è analoga all'equazione 1.4 per l'energia interna e per questo S_{RC} è anche chiamata *densità d'entropia interna*. A questo punto, dato che S_{RC} è analoga all'energia

[4] La densità di energia libera è analoga all'energia libera di Helmholtz vista nel capitolo 1. Infatti, si noti che l'argomento del logaritmo in equazione 4.30 è la funzione di partizione di $P_{RC}(\vec{\sigma}^\star)$.

interna e Φ_{RC} è analoga all'energia libera, è possibile definire la *densità d'entropia esterna* mediante un'equazione simile a 1.5 [5]:

$$\Sigma_{RC} \equiv \Sigma_{RC}(D, y, N, \beta) = \Phi_{RC}(D, y, N, \beta) - yS_{RC}(D, y, N, \beta) \ . \tag{4.33}$$

La densità di entropia esterna Σ_{RC} calcola quindi il logaritmo (diviso per N) del numero di configurazioni di riferimento $\vec{\sigma}^{\star}$ associate a y e D.

La distribuzione ripesata e vincolata nel limite di grandi valori per y e di piccoli valori per D descrive regioni che massimizzano la densità di entropia locale $S_{RC}(D, y, N, \beta)$.

4.3.2 Distribuzione ripesata e non-vincolata

La distribuzione di probabilità *ripesata* e *non-vincolata* P_{RU} è invece espressa da:

$$P_{RU}(\vec{\sigma}^{\star}) = \frac{e^{yNS(\vec{\sigma}^{\star})}}{Z} \ . \tag{4.34}$$

Anche in questo caso è possibile definire la densità di energia libera vincolata Φ_{RU}:

$$\Phi_{RU} \equiv \Phi_{RU}(D, y, N, \beta) = \frac{1}{N} \left\langle ln \sum_{\vec{\sigma^{\star}}} e^{yNS(\vec{\sigma}^{\star})} \right\rangle_{\{\xi^{\mu}\},\{\tau^{\mu}\}}, \tag{4.35}$$

la densità d'entropia locale o densità d'entropia interna S_{RU}:

$$S_{RU} \equiv S_{RU}(D, y, N, \beta) = \frac{\partial}{\partial y}\Phi_{RU}(D, y, N, \beta) = \left\langle \sum_{\{\vec{\sigma}^{\star}\}} P_{RU}(\vec{\sigma}^{\star})\, S(\vec{\sigma}^{\star}) \right\rangle_{\{\xi^{\mu}\},\{\tau^{\mu}\}}, \tag{4.36}$$

e la densità d'entropia esterna Σ_{RU}:

$$\Sigma_{RU} \equiv \Sigma_{RU}(D, y, N, \beta) = \Phi_{RU}(D, y, N, \beta) - yS_{RU}(D, y, N, \beta) \ . \tag{4.37}$$

A differenza di P_{RC}, in questo caso non esiste una corrispondenza con la distribuzione di Boltzmann in equazione 4.24. La distribuzione P_{RU} si ottiene rimuovendo dalla distribuzione P_{RC} ripesata e vincolata in equazione 4.29 la grandezza $\chi(\vec{\sigma^{\star}})$. L'assenza di tale grandezza permette alla distribuzione P_{RU} di accedere alle configurazioni del sistema che non hanno energia nulla. Seppur analiticamente ciò non permetta lo studio delle proprietà delle soluzioni, dal punto di vista pratico questa accessibilità favorisce la creazione di algoritmi d'apprendimento.

Partendo da tale distribuzione, si potrebbe impostare una strategia che selezioni inizialmente una configurazione che non è soluzione e che esegua un procedimento atto a massimizzare la densità d'entropia locale S_{RU}. Questo procedimento è alternativo

[5] Si noti infatti che l'equazione $F = U - TS$ può essere riespressa come: $S = k_B ln\, Z - \frac{U}{T}$. Siccome nel nuovo sistema k_B è una costante formale che si può fissare ad 1 e $y = \frac{1}{T}$ si ottiene: $S = ln\, Z - yU$.

alla minimizzazione dell'energia e cerca di cogliere l'efficacia degli algoritmi euristici efficaci facendo leva sulle proprietà delle soluzioni sub-dominanti. Il fatto che questi algoritmi abbiano performance computazionali migliori suggerisce quindi che la densità di entropia locale non sia pervasa da minimi locali ma sia caratterizzata da un andamento più morbido.

4.3.3 Analisi al non-equilibrio

Lo studio delle proprietà delle soluzioni per questi due tipi di distribuzioni al non-equilibrio segue dei passaggi analoghi all'analisi del potenziale di Franz-Parisi. Questi passaggi, però, non sono eseguiti per analizzare direttamente la densità di entropia locale bensì per sviluppare analiticamente la densità d'energia libera vincolata. Dopo aver sviluppato la densità d'energia libera vincolata, è possibile ricondursi alla densità d'entropia locale mediante una derivata (come visto nelle equazioni 4.32 e 4.36).

Nel caso della distribuzione P_{RC} le osservazioni più rilevanti sono riportato di seguito (ottenute nel limite $\beta \to \infty$).

- Assumendo l'ipotesi di RS, per ogni coppia (α, D) esiste $y^\star$ tale per cui $\Sigma_{RC} = 0$ e $\Sigma_{RC} < 0$ per $y > y^\star$. Ciò significa che nel limite $y \to \infty$ si ha, indipendentemente da α e D, $\Sigma_{RC} < 0$.

- Assumendo l'ipotesi di 1-RSB, per $y \to \infty$ si ottiene ancora $\Sigma_{RC} < 0$. Ciò suggerisce che è necessario utilizzare livelli più complessi di RSB.

Le osservazioni per la P_{RU} sono invece le seguenti (sempre per $\beta \to \infty$).

- É necessario usare direttamente ipotesi di RSB perchè con l'ipotesi di RS si manifestano diversi effetti non fisici.

- Anche in questo caso con l'ipotesi di 1-RSB e per $y \to \infty$ si ottiene $\Sigma_{RC} < 0$.

Nonostante queste osservazioni esprimano le difficoltà nel sviluppare il calcolo per la densità d'energia libera vincolata in entrambe le distribuzioni, si può notare che:

- le quantità ottenute sviluppando per P_{RC} con l'ipotesi di RS e con $y = y^\star$, e per P_{RU} con l'ipotesi di 1-RSB e $y \to \infty$, sono pressochè uguali. Ciò suggerisce che entrambe le analisi stiano descrivendo le stesse regioni associate alla massima densità d'entropia locale.

- Per tutti gli $\alpha < \alpha_C$ la densità di entropia locale approccia la curva con α=0 (che rappresenta la situazione ideale in cui ogni configurazione è soluzione) e per D=0 tale densità di entropia è maggiore di 0 (figura 4.2).

- Per grandi D dominano le soluzioni tipiche all'equilibrio e quindi la densità di entropia locale collassa a quella relativa all'equilibrio.

- Per $\alpha < \alpha_U$, dove α_U=0.77, la densità di entropia locale è monotona descrescente in D; mentre per $\alpha_U < \alpha < \alpha_C$, vi sono regioni dove mancano soluzioni o le cui soluzioni hanno densità di entropia locale negativa.

I dettagli sui procedimenti che hanno portato a questi risultati possono essere ritrovati in letteratura [29, 32, 33].

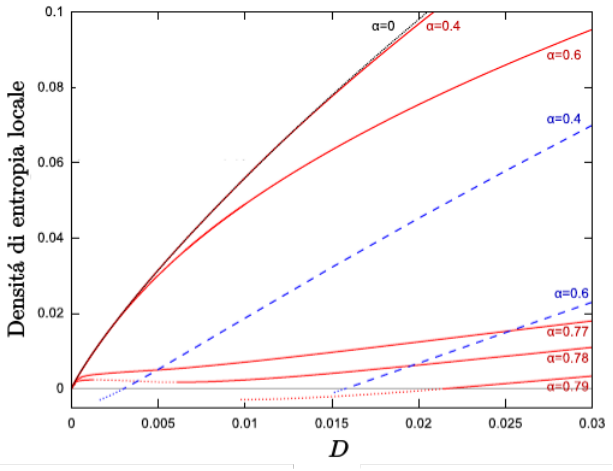

Figura 4.2: Densità di entropia locale vs distanza. La curva in nero è associata al caso ideale in cui ogni configurazione è soluzione (α=0); le curve in rosso rappresentano gli andamenti della densità di entropia locale con l'ipotesi RS (distribuzione RC) e i tratti a puntini segnalano quando tale ipotesi perde validità. Le curve tratteggiate in blu mostrano l'andamento all'equilibrio ed i tratti a puntini identificano situazioni non fisiche. Nonostante non si riescano ad osservare, tutte le curve rosse approcciano la curva ideale per $D \to 0$.

L'impossibilità di utilizzare l'ipotesi di RS per grandi valori di y può essere spiegata da un aumento repentino della complessità dello spazio delle soluzioni. Infatti, se si potesse considerare l'ipotesi di RS anche nel limite $y \to \infty$, ciò indicherebbe l'esistenza di una sola soluzione associata al massimo della densità di entropia locale. Una situazione di questo tipo si descriverebbe ammettendo l'esistenza di un unico cluster denso di soluzioni per y finiti che si riduce progressivamente per y crescenti. Il fatto che questa ipotesi non si possa considerare implica che per valori maggiori di $y^\star$ la struttura dello spazio delle soluzioni ha proprietà più complicate che richiedono l'uso di RSB.

L'esistenza per ogni $\alpha < \alpha_C$ di intorni di $D = 0$ in cui la densità d'entropia è positiva, nonostante alcune curve transino per intervalli negativi di tale grandezza, suggerisce l'esistenza di clusters ricchi di soluzioni con nuclei molto densi.

Il significato di α_U è di grande rilevanza perchè fornisce un criterio per determinare il dominio in cui è gli algoritmi euristici efficaci mostrano ottime prestazioni. É possibile interpretare l'intervallo $\alpha \leq \alpha_U$ come il range in cui gli algoritmi euristici hanno

accesso ad uno spazio caratterizzato da regioni dense di soluzioni, in cui tali soluzioni sub-dominanti sono circondate da un numero esponenziale (in N) di altre soluzioni. Il superamento di questo intervallo, cioè per $\alpha < \alpha_U$, è interpretabile come una transizione da uno spazio dotato di un solo enorme cluster di regioni dense di soluzioni ad uno spazio frammentato di regioni sconnesse isolate.

Quest'ultime ipotesi trovano una conferma numerica. Infatti, il valore stimato analiticamente di α_U per il percettrone binario è circa 0.75 e i valori di capacità massima mostrati dagli algoritmi efficaci oscilla nel range $0.69 < \alpha_C < 0.75$.

4.3.4 Risultati per la generalizzazione

Le osservazioni e i risultati mostrati finora si sono basati sempre sullo studio del percettrone binario nel caso del problema di classificazione. Per quanto il problema di generalizzazione sia ottimamente risolvibile anche nel caso di variabili discrete, è necessario studiarne le proprietà al non-equilibrio per stabilire se anche in questo caso gli effetti fuori dall'equilibrio portino dei benefici.

Un'analisi del potenziale di Franz-Parisi per valutare le proprietà all'equilibrio del potenziale di Franz-Parisi conferma che le soluzioni tipiche sono isolate e che la configurazione del percettrone teacher risulta, oltre che isolata, indistinguibile dalle altre soluzioni.

L'analisi numerica eseguita sulle soluzioni degli algoritmi euristici ha invece evidenziato che:

- le soluzioni algoritmiche generalizzano meglio rispetto alle soluzioni tipiche, ovvero che hanno un minor errore di generalizzazione;
- la stima numerica della densità d'entropia per un sistema vincolato ad una soluzione algoritmica (eccetto la configurazione del percettrone teacher) è diversa dalle previsioni analitiche all'equilibrio.

Questi risultati confermano l'incongruenza con l'analisi all'equilibrio e suggeriscono che si possa giovare dell'utilizzo degli algoritmi euristici efficaci anche nel caso del problema di generalizzazione.

Estendendo le analisi di Franz-Parisi alle distribuzioni P_{RC} e P_{RU} per il problema di generalizzazione, si ritrovano gli stessi risultati ottenuti per il problema di classificazione ed in particolare:

- $\alpha_U \approx 1.1$ (ricordando che la capacità massima in questo caso è $\alpha_{TS} = 1.245$);
- le soluzioni sub-dominanti confermano il minor errore di generalizzazione mostrato dagli algoritmi euristici (figura 4.3).

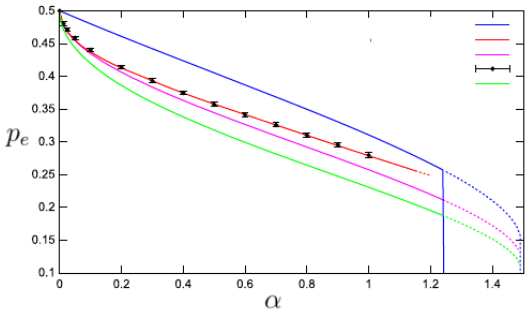

Figura 4.3: Errore di generalizzazione vs capacità. La curva blu è relativa ad una soluzione tipica, la curva rossa ad una soluzione sub-dominante a distanza piccole. La curva magenta è relativa ad una soluzione sub-dominante a più grande distanza, e la curva verde rappresenta l'errore di generalizzazione degli algoritmi Bayesiani [34].

La miglior capacità di generalizzazione è intuitivamente giustificabile dal fatto che una soluzione sub-dominante, essendo immersa in una regione ricca di soluzioni, assume un ruolo rappresentativo della regione stessa. Questo ruolo amplia la capacità di risolvere i pattern non appartenenti al training set.

Capitolo 5

replicated focusing Belief Propagation

Questo capitolo spiega il funzionamento del primo algoritmo di apprendimento sviluppato mediante l'uso della distribuzione di probabilità ripesata e non-vincolata mostrata nel capitolo precedente. Tale algoritmo è chiamato *replicated focusing Belief Propagation* (rfBP) e mira a recuperare l'efficienza degli algoritmi euristici efficaci sfruttando effetti fuori dall'equilibrio.

La prima sezione illustra per semplificità come si possa impostare un algoritmo d'apprendimento a partire dalla distribuzione all'equilibrio (distribuzione di Boltzmann) sfruttando la Belief Propagation.

La seconda sezione mostra che è possibile definire un nuovo sistema, chiamato *Robust Ensemble* (RE), in grado di favorire l'implementazione di algoritmi basati sulla distribuzione al non-equilibrio in equazione 4.34.

La terza sezione descrive il funzionamento di base dell'algoritmo di rfBP evidenziando come l'uso del RE modifichi l'implementazione all'equilibrio.

Infine, la quarta sezione definisce gli elementi che caratterizzano l'algoritmo rfBP, ne analizza i risultati in relazione al primo algoritmo euristico creato e ai risultati analitici delineati nella sezione 4.3.3.

5.1 Implementazione all'equilibrio

In base a quanto detto nel capitolo 4, sono le NNs feed-forward fully-connected discrete le strutture di calcolo su cui si vuole realizzare l'apprendimento. In particolare, si è sempre analizzato per chiarezza il caso del percettrone binario.

In questa sezione sono prese in considerazione due tipi di NNs binarie: il percettrone e le committee machine. Si ricorda che per NNs binarie si intendono NNs con componenti

dei pesi $\vec{\sigma}$, componenti dei vettori di input $\vec{\xi^{\mu}}$ e outputs τ^{μ} che possono assumere i valori ± 1.

Le committee machine sono NNs feed-forward fully-connected con un hidden layer composto da K unità di processing e che producono un singolo output [1]. Seppur anche in letteratura siano principalmente mostrati i risultati per questi due tipi di NNs, sono attesi risultati analoghi anche per NNs discrete (non necessariamente binarie) con più di un singolo hidden layer.

Il tipo di implementazione mostrato di seguito è valido sia per il problema di classificazione che per il problema di generalizzazione. Nonostante ciò, si è scelto per convenienza di far sempre riferimento al problema di classificazione, i cui tratti sono stati esposti in sezione 4.1.1.

5.1.1 Percettrone binario

Come mostrato in equazione 4.5, l'energia del sistema che descrive il percettrone binario è uguale a:

$$E(\vec{\sigma}) = \sum_{\mu=1}^{M} \Theta\Big(- \tau^{\mu} \sum_{i=1}^{N} \sigma_i \, \xi_i^{\mu} \Big), \tag{5.1}$$

dove la sommatoria $\sum_{i=1}^{N} \sigma_i \, \xi_i^{\mu}$ non è altro che il prodotto scalare $\vec{\sigma} \cdot \vec{\xi^{\mu}}$.

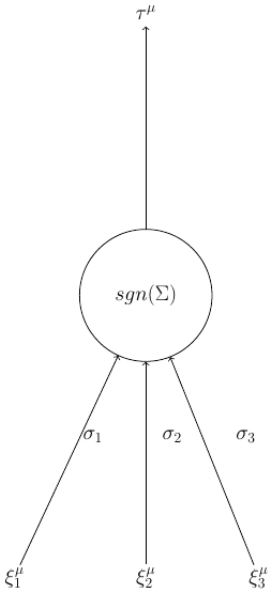

Figura 5.1: Esempio di un percettrone con N=3, il simbolo $sgn(\Sigma)$ indica le due operazioni di calcolo effettuate.

[1] Si noti che nel caso in cui $K = 1$, la rete è ridotta alla struttura del percettrone.

Seguendo i discorsi nella sezione 4.1.1, l'apprendimento sul percettrone binario può essere anche interpretato definendo la quantità:

$$\chi(\vec{\sigma}) = \prod_{\mu=1}^{M} \Theta\Big(\tau^{\mu} \sum_{i=1}^{N} \sigma_i \, \xi_i^{\mu}\Big) \tag{5.2}$$

e determinando un $\vec{\sigma}$ tale per cui $\chi(\vec{\sigma}) = 1$. Questa quantità consente di esprimere, nel limite $\beta \to \infty$, la distribuzione di Boltzmann come (equazione 4.24):

$$P(\vec{\sigma}) = \frac{\chi(\vec{\sigma})}{Z} \ , \tag{5.3}$$

che è diversa da 0 solo per le configurazioni che sono soluzioni del problema $\vec{\sigma} = \vec{\sigma}^{\star}$. Utilizzando al numeratore il termine destro dell'equazione 5.2, la distribuzione di probabilità di Boltzmann si può riscrivere come una funzione fattorizzata:

$$P(\vec{\sigma}) = \frac{1}{Z} \prod_{\mu=1}^{M} \Theta(\tau^{\mu} \, \tau(\vec{\sigma}, \vec{\xi^{\mu}})) \ . \tag{5.4}$$

Di conseguenza, la distribuzione è rappresentabile in un grafo fattoriale (sezione 1.3), dotato di un nodo fattore per ogni $\Theta\Big(\tau^{\mu} \, (\vec{\sigma} \cdot \vec{\xi^{\mu}})\Big)$ e di un nodo variabile per ogni σ_i. Oltre ad essi, è conveniente inserire nel grafo anche un nodo relativo ai valori in output $\{\tau^{\mu}\}_{\mu=1}^{M}$, trattando tali valori come variabili alla pari dei pesi, ed aggiungendo dei nodi fattore extra che vincolino questi nuovi nodi ad assumere il loro valore noto. La figura 5.2 riporta il grafo fattoriale associato ad un percettrone allenato su un solo pattern.

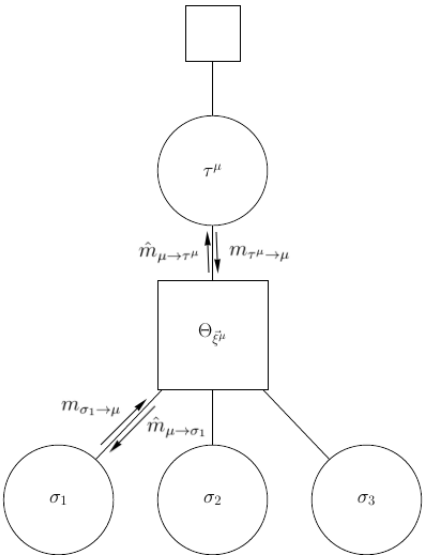

Figura 5.2: Grafo fattoriale associato ad un percettrone con $N = 3$ per un pattern $\vec{\xi}^{\mu}$ generico.

La figura 5.3 mostra invece il grafo associato ad un percettrone allenato su più di un pattern. Si può osservare che per ogni $\vec{\xi}^{\mu}$ esiste il corrispondente nodo fattore $\Theta\Big(\tau^{\mu}\,(\vec{\sigma}\cdot\vec{\xi}^{\mu})\Big)$.

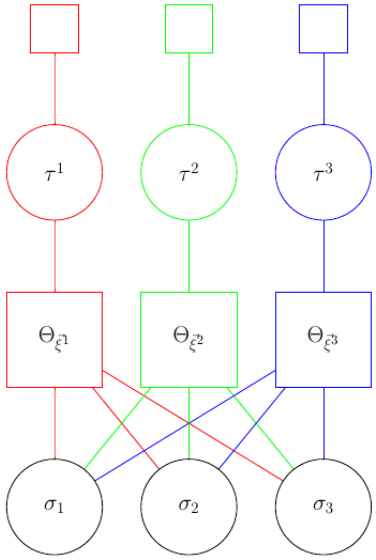

Figura 5.3: Grafo fattoriale associato ad un percettrone con $N = 3$ per i tre pattern $\vec{\xi}^1$, $\vec{\xi}^2$ e $\vec{\xi}^3$ distinti dal colore.

A questo punto, sul grafo è possibile eseguire l'algoritmo di Belief Propagation per calcolare le probabilità marginali $P_i(\sigma_i)$ (sezione 2.2.2). I messaggi inviati dai nodi variabile σ_i e τ^{μ} ai nodi fattori $\Theta\Big(\tau^{\mu}\sum_{i=1}^{N}\sigma_i\,\xi_i^{\mu}\Big)$, sintetizzati con μ, sono quindi descritti dalla prima equazione in 3.24. Invece, i messaggi inviati dal nodi fattore μ ai nodi variabile σ_i e all'output τ^{μ} sono espressi dalla seconda equazione in 3.24. Una volta raggiunta la convergenza per tali messaggi è possibile stimare i pesi σ_i determinando quale valore tra ± 1 massimizza $P_(\sigma_i)$ [2] .

I motivi per cui si possano ricavare solo delle stima dei pesi e non il loro valore esatto sono due. Il primo è il fatto che il training su più pattern genera dei grafi fattoriali che *non sono ad albero* (come si può vedere nell'esempio in figura 5.3). Nonostante ciò, è stato dimostrato che per N grandi le variabili nei loop presenti sono debolmente correlate e quindi la BP calcola delle approssimazioni con valori prossimi ai risultati esatti. Il secondo motivo è la necessità computazionale di dover fissare un limite massimo di iterazioni all'algoritmo BP.

Per comodità, i messaggi della BP possono essere parametrizzati analogamente a quanto fatto per ricavare le magnetizzazioni cave nel capitolo 3 (equazioni 3.26). Di

[2]Si noti che, come detto nel capitolo 2, le distribuzione marginali stimate dalla BP sono non normalizzate.

conseguenza i messaggi dai nodi variabili ai nodi fattore diventano:

$$\begin{aligned} m^t_{\sigma_i\to\mu} &= tanh\Bigg[\sum_{k\in\partial\sigma_i\setminus\mu} artanh(\hat{m}^t_{k\to\sigma_i})\Bigg] \\ m^t_{\tau^\mu\to\mu} &= tanh\Bigg[\sum_{k\in\partial\tau^\mu\setminus\mu} artanh(\hat{m}^t_{k\to\tau^\mu})\Bigg] \end{aligned}. \tag{5.5}$$

Diversamente, i messaggi inviati dai nodi fattore ai nodi variabile non possono essere semplificati univocamente, vista la presenza della funzione Θ, e per questo si riportano puramente le rispettive parametrizzazioni:

$$\begin{aligned} \hat{m}^t_{\mu\to\sigma_i} &= \frac{\sum_{\sigma_i}\sigma_i\,\hat{\nu}^t_{\mu\to\sigma_i}(\sigma_i)}{\sum_{\sigma_i}\hat{\nu}^t_{\mu\to\sigma_i}(\sigma_i)} \\ \hat{m}^t_{\mu\to\tau^\mu} &= \frac{\sum_{\tau^\mu}\tau^\mu\,\hat{\nu}^t_{\mu\to\tau^\mu}(\tau^\mu)}{\sum_{\tau^\mu}\hat{\nu}^t_{\mu\to\tau^\mu}(\tau^\mu)}\ , \end{aligned} \tag{5.6}$$

dove i messaggi $\hat{\nu}^t_{\mu\to\sigma_i}(\sigma_i)$ e $\hat{\nu}^t_{\mu\to\tau^\mu}(\tau^\mu)$ sono rispettivamente uguali a:

$$\begin{aligned} \hat{\nu}^{t+1}_{\mu\to\sigma_i}(\sigma_i) &= \sum_{\tau^\mu,\{\sigma_j\}_{j\neq i}}\Big[\Theta\Big(\tau^\mu\Big(\sum_{l\in n(\mu)}\sigma_l\,\xi^\mu_l\Big)\Big)\Big(\frac{1+\tau^\mu m^t_{\tau^\mu\to\mu}}{2}\Big)\prod_{k\in n(\mu)\setminus i}\Big(\frac{1+\sigma_k m^t_{\sigma_k\to\mu}}{2}\Big)\Big] \\ \hat{\nu}^{t+1}_{\mu\to\tau^\mu}(\tau^\mu) &= \sum_{\{\sigma_j\}}\Big[\Theta\Big(\tau^\mu\Big(\sum_{l\in n(\mu)}\sigma_l\,\xi^\mu_l\Big)\Big)\prod_{k\in n(\mu)}\Big(\frac{1+\sigma_k m^t_{\sigma_k\to\mu}}{2}\Big)\Big]. \end{aligned} \tag{5.7}$$

Si ricordi che il "tempo" t indicizza gli step di esecuzione e serve ad evidenziare l'aggiornamento iterativo dei messaggi.

Utilizzando i messaggi parametrizzati, si perde la dipendenza dei messaggi dalle variabili σ_i e, una volta raggiunta la convergenza, i pesi si possono determinare dal segno delle m_{σ_i}, interpretabili come magnetizzazioni totali. Infatti, l'equazione delle magnetizzazioni totali potrebbe essere anche scritta:

$$m_i = \frac{\sum_{\sigma_i}\sigma_i P_i(\sigma_i)}{\sum_{\sigma_i}P_i(\sigma_i)}, \tag{5.8}$$

che coincide con:

$$m^t_i = \frac{\sum_{\sigma_i}\sigma_i \nu^t_i(\sigma_i)}{\sum_{\sigma_i}\nu^t_i(\sigma_i)}, \tag{5.9}$$

quando è raggiunta la convergenza. Le magnetizzazioni totali m_i rappresentano quindi una sorta di valore d'aspettazione per i singoli pesi σ_i, che di conseguenza possono essere stimati da $\sigma_i = sgn(m_i)$.

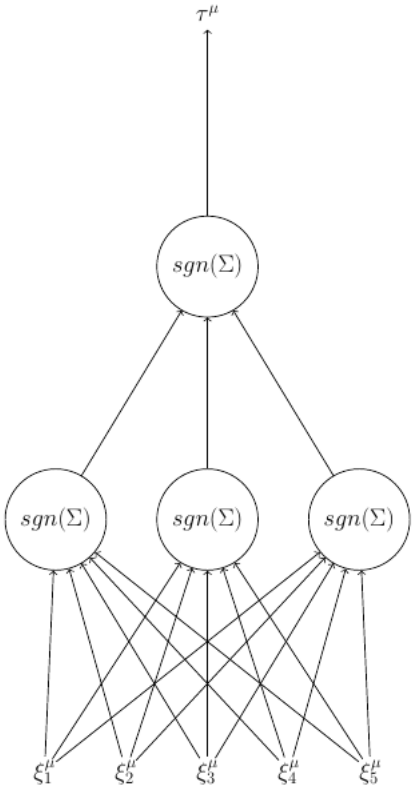

Figura 5.4: Esempio di committee machine con K=3 ed N=15.

5.1.2 Committee Machine

Le committee machine (CM) sono NNs feed-forwards fully-connected costituite da un hidden layer di K unità in cui ogni unità $k \in K$ riceve lo stesso vettore di input [35]. Si definiscono pesi $\vec{\sigma}$ delle CM come i pesi delle sinapsi che collegano le componenti del vettore di input alle unità K del singolo hidden layer. I pesi delle sinapsi che collegano l'hidden layer al valore in output sono fissati convenzionalmente tutti a 1.

Le committe machine producono per un generico vettore di input $\vec{\xi}^\mu$ un valore in output pari a:

$$\tau^\mu = sgn\Big(\sum_k sgn(\vec{\sigma}_{k=1}^K \cdot \vec{\xi}^\mu)\Big) \ , \tag{5.10}$$

dove i $\vec{\sigma}_k$ sono subsets dell'intero set di pesi $\vec{\sigma}$ e rappresentano i pesi delle sinapsi associate all'unità hidden k.

Analogamente al caso del percettrone binario, è possibile definire l'energia delle CM binarie (per il problema di classificazione):

$$E(\vec{\sigma}) = \sum_{\mu=1}^{M} \Theta\Big(- \tau^\mu \sum_{k=1}^{K} sgn(\vec{\sigma}_k \cdot \vec{\xi}^\mu)\Big) \tag{5.11}$$

e la quantità:

$$\chi(\vec{\sigma}) = \prod_{\mu=1}^{M} \Theta\Big(\tau^{\mu} \sum_{k=1}^{K} sgn(\vec{\sigma}_k \cdot \vec{\xi}^{\mu})\Big). \tag{5.12}$$

Lo scopo dell'apprendimento può essere anche in questo caso interpretato o come la minimizzazione dell'energia o come la ricerca di un $\vec{\sigma}$ tale per cui $\chi(\vec{\sigma}) = 1$.

Ripercorrendo pressochè gli stessi passaggi adottati per il caso del percettrone, partendo dalla distribuzione di Boltzmann, si ritrova una distribuzione di probabilità all'equilibrio per $\beta \to \infty$ uguale a:

$$P(\vec{\sigma}) = \frac{1}{Z} \prod_{\mu=1}^{M} \Theta\Big(\tau^{\mu} \sum_{k=1}^{K} sgn(\vec{\sigma}_k \cdot \vec{\xi}^{\mu})\Big) = \frac{1}{Z}\chi(\vec{\sigma}), \tag{5.13}$$

dove la funzione di partizione è uguale a:

$$Z = \sum_{\vec{\sigma}} \prod_{\mu=1}^{M} \Theta\Big(\tau^{\mu} \sum_{k=1}^{K} sgn(\vec{\sigma}_k \cdot \vec{\xi}^{\mu})\Big) = \sum_{\vec{\sigma}} \chi(\vec{\sigma}). \tag{5.14}$$

Analogamente al percettrone, anche il sistema definito per le CM può essere rappresentato mediante i grafi fattoriali.

Il grafo fattoriale rappresentabile dall'equazione 5.13 è praticamente uguale a quello per il percettrone in 5.2 se non per funzioni Θ con un argomento più elaborato. In modo da mantenere una consistenza maggiore con il percettrone e per passare in maniera più intuitiva dal grafico della committee machine al grafo fattoriale associato, si valuti la committee machine da una diversa prospettiva.

Rappresentazioni interne

I nodi hidden possono essere interpretati come singoli percettroni che producono valori in output intermedi uguali a $sgn(\vec{\sigma}_k \cdot \vec{\xi}^{\mu})$ [36]. Nonostante questi valori intermedi siano già conosciuti, si possono aggiungere delle variabili ausiliare che li rappresentino. Si definiscano quindi tali variabili ausiliarie come *rappresentazioni interne* h_k^{μ}. Le h_k^{μ} hanno sempre natura binaria e si possono considerare come una rappresentazione degli input $\vec{\xi}^{\mu}$ generata dall'hidden layer. Si osservi in figura 5.5 un esempio di come diventa una CM all'inserimento delle rappresentazioni interne.

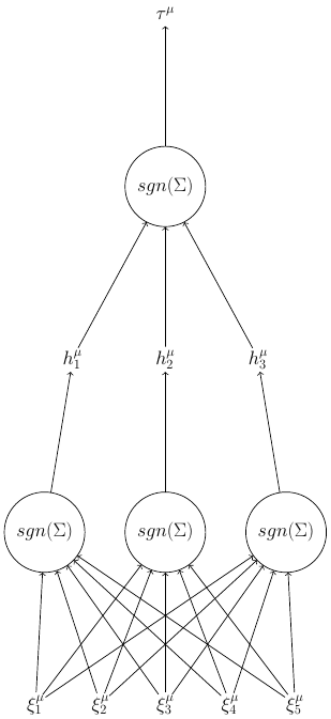

Figura 5.5: Esempio di una committee con K=3 ed N=15 con l'inserimento delle rappresentazioni interne.

A questo punto, l'equazione 5.12 diventa:

$$\chi(\vec{\sigma}, \{\vec{h}^\mu\}) = \prod_{\mu=1}^{M} \Theta\Big(\tau^\mu \sum_{k=1}^{K} h_k^\mu\Big) \prod_{k=1}^{K} \Theta\Big(h_k^\mu \, sgn(\vec{\sigma}_k \cdot \vec{\xi}^\mu)\Big) \, , \tag{5.15}$$

e la risoluzione del problema di classificazione necessita di ritrovare dei pesi $\vec{\sigma}$ e delle rappresentazioni interne $\{\vec{h}^\mu\}_{\mu=1}^M$ tali per cui: $\chi(\vec{\sigma}, \{\vec{h}^\mu\}_{\mu=1}^M) = 1$.
Questo nuovo sistema è equivalente all'originale, visto che le rappresentazioni interne sono vincolate ad essere gli output dell'hidden layer, e le due quantità in 5.12 e 5.15 si corrispondono

La distribuzione di probabilità in 5.13 può quindi essere riespressa equivalentemente come:

$$P(\vec{\sigma}, \{\vec{h}^\mu\}) = \frac{1}{Z} \prod_{\mu=1}^{M} \Theta\Big(\tau^\mu \sum_{k=1}^{K} h_k^\mu\Big) \prod_{k=1}^{K} \Theta\Big(h_k^\mu \, sgn(\vec{\sigma}_k \cdot \vec{\xi}^\mu)\Big). \tag{5.16}$$

A questa distribuzione di probabilità corrisponde il grafo fattoriale in figura 5.6. In questo grafo esiste un nodo variabile per ogni σ_i, h_k^μ e τ^μ, e un nodo fattore per ogni $\Theta\Big(\tau^\mu \sum_k h_k^\mu\Big)$ e $\Theta\Big(h_k^\mu \, sgn(\vec{\sigma}_k \cdot \vec{\xi}^\mu)\Big)$. Per semplicità i nodi fattori sono indicati rispettivamente come $\Theta_{\vec{\xi}^\mu}$

e Θ_μ. Come per il percettrone, esiste una nodo fattore distinto per ogni diverso pattern ed un esempio di ciò è visualizzabile nella seguente figura 5.7.

Nel grafo delle CM la Belief Propagation è eseguita trattando l'intero grafo come tanti sottografi fattoriali associati ad un percettrone (figura 5.2). Di conseguenza i messaggi scambiati in questo grafo hanno esattamente la stessa forma di quelli visti per il percettrone in 5.1.1.

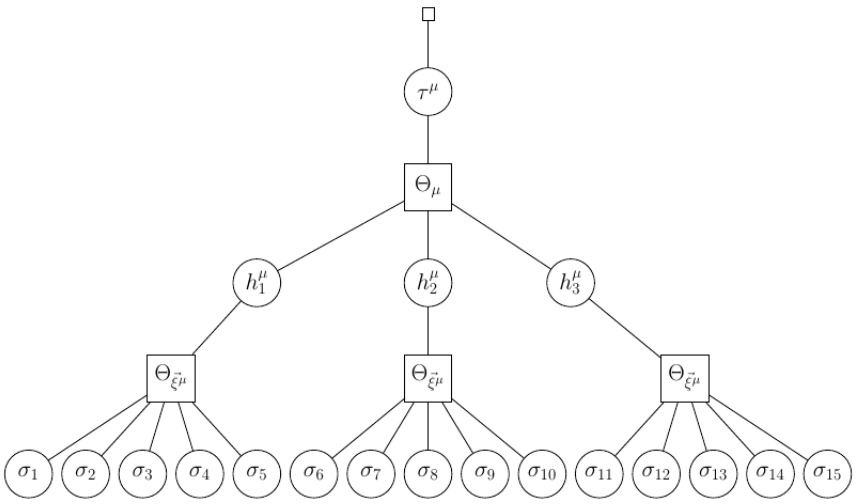

Figura 5.6: Grafo fattoriale associato ad una committee machine con $K = 3$ e $N = 15$, con l'inserimento delle rappresentazioni interne, per un pattern generico $\vec{\xi}^\mu$.

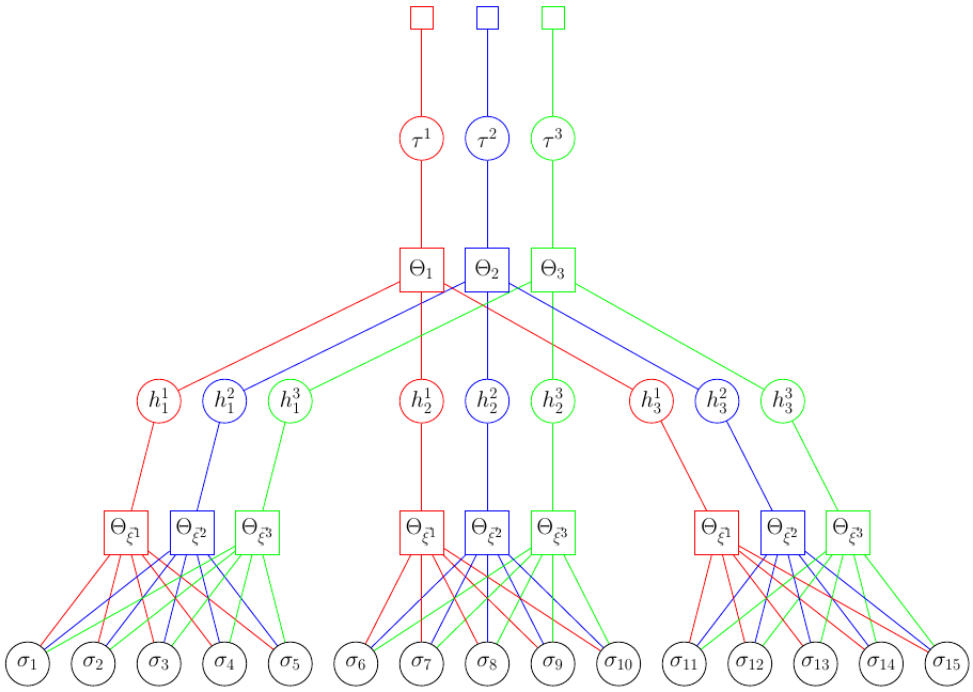

Figura 5.7: Grafo fattoriale associato ad una committee machine con $K = 3$ e $N = 15$, con l'inserimento delle rappresentazioni interne, per i tre pattern $\vec{\xi}^1$, $\vec{\xi}^2$ e $\vec{\xi}^3$ distinti dal colore.

5.2 Robust Ensemble

La distribuzione di probabilità ripesata e non-vincolata, in sezione 4.3.2, è una distribuzione al non-equilibrio ideata per fornire una base di sviluppo di algoritmi d'apprendimento capaci di sfruttare effetti fuori dall'equilibrio. Questa distribuzione, espressa da:

$$P_{RU}(\vec{\sigma}^\star) = \frac{e^{yNS(\vec{\sigma}^\star)}}{\sum_{\vec{\sigma}^{\star\prime}} e^{yNS(\vec{\sigma}^{\star\prime})}} \tag{5.17}$$

ha una funzione di partizione uguale a:

$$Z \equiv Z(y, D, \beta) = \sum_{\vec{\sigma}^\star} e^{yNS(\vec{\sigma}^\star)} \tag{5.18}$$

in cui $S(\vec{\sigma}^\star)$ rappresenta la densità di entropia locale per un sistema vincolato a $\vec{\sigma}^\star$ (in 4.20):

$$S(\vec{\sigma}^\star) = \frac{1}{N} ln\left(\sum_{\vec{\sigma}} e^{-\beta E(\vec{\sigma})} \delta\Big(d(\vec{\sigma}, \vec{\sigma^\star}), D\Big)\right). \tag{5.19}$$

Ora, come indicato nella sezione 4.2.1, il vincolo dato dal delta di Kroenecker si può sostituire per semplicità con la funzione $e^{\gamma\vec{\sigma}\cdot\vec{\sigma}^\star}$. Sostituendo, $S(\vec{\sigma}^\star)$ diventa:

$$S(\vec{\sigma}^\star) = \frac{1}{N} ln\left(\sum_{\vec{\sigma}} e^{-\beta E(\vec{\sigma}) + \gamma\vec{\sigma}\cdot\vec{\sigma}^\star} \right) \tag{5.20}$$

e la funzione di partizione del sistema diventa:

$$Z \equiv Z(y, \gamma, \beta) = \sum_{\vec{\sigma}^\star} e^{yNS(\vec{\sigma}^\star)}. \tag{5.21}$$

Tale sostituizione stabilisce una condizione meno rigida e semplifica la trattazione, dato che aggiunge all'energia del sistema una componente nella stessa forma dell'Hamiltoniana dei problemi di Spin Glass (equazione 3.9) [3]. Il parametro γ regola il peso da attribuire alla distanza tra le configurazioni: maggiore è il suo valore, maggiore è il peso assegnato alle configurazioni più vicine alla configurazione $\vec{\sigma}^\star$ di riferimento.

A questo punto, se y è un intero non negativo, la funzione di partizione associata a 5.21 può essere riorganizzata in questo modo (dettagli in appendice E):

$$Z = \sum_{\vec{\sigma}^\star} \sum_{\{\vec{\sigma}^a\}} e^{-\beta \sum_{a=1}^{y} E(\vec{\sigma}^a) + \gamma \sum_{a=1}^{y} \vec{\sigma}^a\cdot\vec{\sigma}^\star}. \tag{5.23}$$

Questa grandezza è la funzione di partizione di un sistema in cui sono presenti in totale $y+1$ repliche del sistema. Una tra queste repliche, $\vec{\sigma}^\star$, è assunta di riferimento mentre le rimanenti y (indicizzate da a in 5.23) sono identiche e, oltre ad interagire con $\vec{\sigma}^\star$, sono hanno energia $E(\vec{\sigma}^a)$. Si noti che risolvere la sommatoria [4] $\sum_{\{\vec{\sigma}^a\}}$ significa riottenere il sistema di partenza descritto dalla funzione di partizione 5.21. Diversamente, risolvere $\sum_{\vec{\sigma}^\star}$ significa trovare la funzione di partizione di un sistema formato dalle restanti y repliche. Il sistema delle repliche $\{\vec{\sigma}^a\}_{a=1}^{y}$ è definito *Robust Ensemble* [37] ed è descritto dalla seguente funzione di partizione:

$$Z \equiv Z(\beta, y, \gamma) = \sum_{\{\vec{\sigma}^a\}} e^{-\beta(\sum_{a=1}^{y} E(\vec{\sigma}^a) + A(\{\vec{\sigma}^a\}))} \ . \tag{5.24}$$

Il termine $A(\{\vec{\sigma}^a\}_{a=1}^{y}, \gamma, \beta)$ è il termine analitico relativo alla risoluzione della sommatoria $\sum_{\vec{\sigma}^\star}$:

$$A(\{\vec{\sigma}^a\}) = -\frac{1}{\beta} ln \sum_{\vec{\sigma}^\star} e^{\gamma \sum_{a=1}^{y} \vec{\sigma}^a\cdot\vec{\sigma}^\star} \ . \tag{5.25}$$

[3] Si noti infatti, che il termine all'esponente in 5.20 relativo a γ si può esprimere come:

$$E(\vec{\sigma}, \vec{\sigma}^\star) = \sum_{i=1}^{N} \gamma\sigma_i\, \sigma_i^\star. \tag{5.22}$$

[4] $\sum_{\{\vec{\sigma}^a\}} = \sum_{\vec{\sigma}^1} \sum_{\vec{\sigma}^2} .. \sum_{\vec{\sigma}^y}$

5.3 Implementazione al non-equilibrio

Ora, si aggiunga il Robust Ensemble al sistema delle configurazioni di riferimento $\vec{\sigma}^\star$. La funzione di partizione di questo nuovo sistema è formulata dall'equazione 5.23 e la distribuzione di probabilità del sistema è:

$$P(\vec{\sigma}^\star, \{\vec{\sigma}^a\}) = \frac{1}{Z} e^{-\beta \sum_{a=1}^{y} E(\vec{\sigma}^a) + \gamma \sum_{a=1}^{y} \vec{\sigma}^a \cdot \vec{\sigma}^\star} \ , \tag{5.26}$$

anche esprimibile come:

$$P(\vec{\sigma}^\star, \{\vec{\sigma}^a\}) = \frac{1}{Z} \prod_{a=1}^{y} e^{-\beta E(\vec{\sigma}^a) + \gamma \vec{\sigma}^a \cdot \vec{\sigma}^\star}. \tag{5.27}$$

Sviluppando la distribuzione nel limite $\beta \to \infty$ e considerando sempre NNs binarie, si ottiene:

$$P(\vec{\sigma}^\star, \{\vec{\sigma}^a\}) = \frac{1}{Z} \prod_{a=1}^{y} \chi(\vec{\sigma}^a) e^{\gamma \vec{\sigma}^a \cdot \vec{\sigma}^\star}. \tag{5.28}$$

Inoltre, svolgendo il prodotto scalare $\vec{\sigma}^a \cdot \vec{\sigma}^\star = \sum_{i=1}^{N} \sigma_i^a \, \sigma_i^\star$, la distribuzione diventa:

$$P(\vec{\sigma}^\star, \{\vec{\sigma}^a\}) = \frac{1}{Z} \prod_{a=1}^{y} \chi(\vec{\sigma}^a) \prod_{i=1}^{N} e^{\gamma \sigma_i^a \, \sigma_i^\star}. \tag{5.29}$$

Quest'ultima distribuzione è stata ottenuta indipendentemente dalla struttura di una NN binaria. Per il percettrone e per le CM, $P(\vec{\sigma}^\star, \{\vec{\sigma}^a\})$ diventa rispettivamente uguale a:

$$P(\vec{\sigma}^\star, \{\vec{\sigma}^a\}) = \frac{1}{Z} \prod_{a=1}^{y} \prod_{\mu=1}^{M} \Theta\Big(\tau^\mu \sum_{i=1}^{N} \sigma_i^a \, \xi_i^\mu\Big) \prod_{i=1}^{N} e^{\gamma \sigma_i^a \, \sigma_i^\star} \tag{5.30}$$

e

$$P(\vec{\sigma}^\star, \{\vec{h}^\mu\}, \{\vec{\sigma}^a\}) = Z(y,\gamma)^{-1} \prod_{a=1}^{y} \prod_{\mu=1}^{M} \Theta\big(\tau^\mu \sum_{k=1}^{K} h_k^\mu\big) \prod_{k=1}^{K} \Theta\big(h_k^\mu \, sgn(\vec{\sigma}_k^a \cdot \vec{\xi}^\mu)\big) \prod_{i=1}^{N} e^{\gamma \sigma_i^a \, \sigma_i^\star}. \tag{5.31}$$

Si può osservare che per la CM sono state usate le rappresentazioni interne h_k^μ.

Queste due distribuzioni continuano ad essere fattorizzatie e quindi rappresentabili mediante i grafi fattoriali. Nello specifico, i nuovi grafi sono costituti da y repliche dei singoli grafi visti in precedenza (rispettivamente figura 5.2 e 5.6), dato che esiste un grafo per ciascuna replica $\vec{\sigma}^a$. Oltre a ciò, ogni replica $\vec{\sigma}^a$ interagisce con la replica di riferimento $\vec{\sigma}^\star$ tramite il termine $e^{\gamma \vec{\sigma}^a \cdot \vec{\sigma}^\star}$. Tale interazione è realizzata nel grafo fattoriale aggiungendo un nodo fattore γ per ogni funzione $e^{\gamma \sigma_i^a \, \sigma_i^\star}$ in 5.31 e collegando i nodi variabile σ_i^a e $\sigma_i^\star$ a tale nodo. In questa maniera tutte le repliche dello stesso peso σ_i sono indirettamente connesse dai nodi fattore γ, come si può visualizzare nell'esempio in figura 5.8.

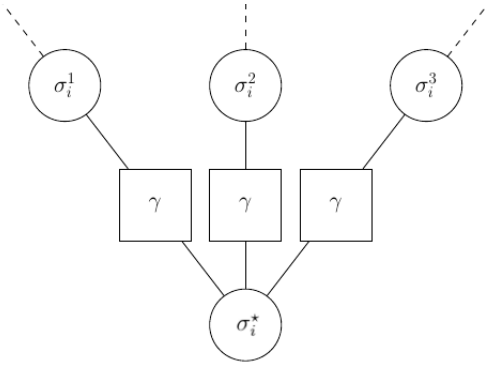

Figura 5.8: Connessione di tre repliche identiche del peso σ_i con la replica di riferimento.

5.3.1 Relazione tra le repliche

In questi nuovi grafi fattoriali i messaggi di Belief Propagation continuano ad essere formulati dalle equazioni mostrate in sezione 5.1.1. Una modifica dei messaggi per i pesi replicati σ_i^a è però inevitabile, visto che essi sono connessi mediante i fattori γ al riferimento e quindi indirettamente anche a tutte le altre repliche.

Come spiegato dall'equazione E.2, tutte le y repliche del sistema sono identiche nel senso che

$$\sum_{\vec{\sigma}^a} e^{-\beta E(\vec{\sigma}^a) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^a)} \tag{5.32}$$

è uguale per $\forall a$. In altre parole, data una configurazione di riferimento $\vec{\sigma}^\star$, ogni singola replica è descritta dalla distribuzione:

$$P(\vec{\sigma}^a) = \frac{1}{Z} e^{-\beta E(\vec{\sigma}^a) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^a)} \ , \tag{5.33}$$

e la funzioni di partizione $Z \equiv Z(\vec{\sigma}^\star, \gamma, \beta)$ è uguale $\forall a$. Questo tipo di identità indica che ogni replica del sistema può potenzialmente realizzarsi su una configurazione $\vec{\sigma}^a$ diversa per ogni a.

Senza stabilire un'ipotesi di relazione tra tali repliche, la regola di aggiornamento della BP dovrebbe essere applicata contemporaneamente sull'intero insieme connesso degli y grafi. Ciò significherebbe appesantire l'esecuzione dell'algoritmo ed aver una gestione limitata di y. É necessario quindi assumere un'ipotesi sulle repliche del RE che renda efficace l'integrazione della Belief Propagation.

In letteratura, l'ipotesi euristica che permette di creare un risolutore efficace è supporre che i messaggi scambiati nei grafi delle singole repliche del sistema siano gli stessi. I messaggi inviati dai singoli pesi replicati σ_i^a al riferimento $\sigma_i^\star$ e viceversa sono quindi tutti uguali (figura 5.9) e ciò permette di semplificare la trattazione degli y grafi fattoriali ad un singolo grafo, a cui è aggiunto un nodo extra che invia il contributo delle restanti repliche. Questo passaggio è evidenziato dalle figure 5.9 e 5.10.

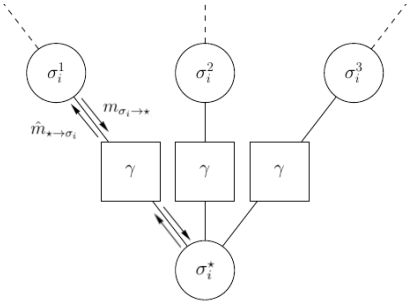

Figura 5.9: Connessione di tre repliche del peso σ_i.

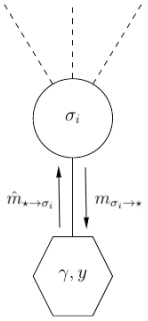

Figura 5.10: Semplificazione del contributo dalle repliche per il peso σ_i.

Nel caso del percettrone e delle CM i grafi fattoriali integrati con il RE sono illustrati rispettivamente in figura 5.11 ed in figura 5.12.

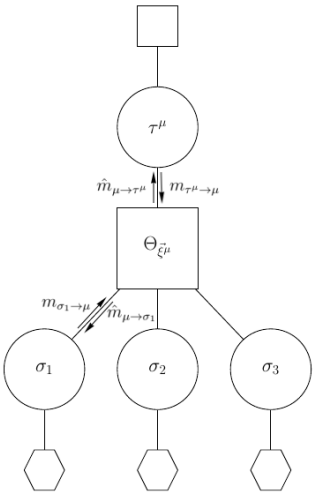

Figura 5.11: Esempio di grafo fattoriale con repliche associato ad un percettrone con $N = 3$ per un pattern generico $\vec{\xi}^\mu$.

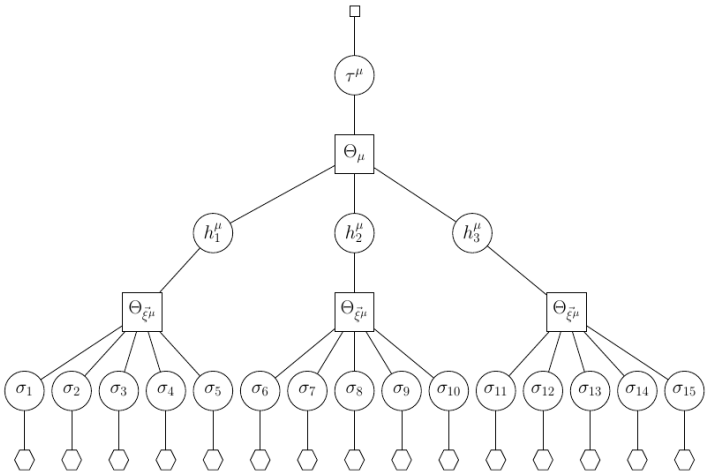

Figura 5.12: Esempio di grafo fattoriale con repliche associato ad una committee machine con $N = 15$ e $K = 3$ e con le rappresentazioni interne per un pattern generico $\vec{\xi}^\mu$.

Per completezza, si riportano nelle figure 5.13 e 5.14 come diventano i grafi del percettrone e delle CM quando sono allenati su più di un solo pattern.

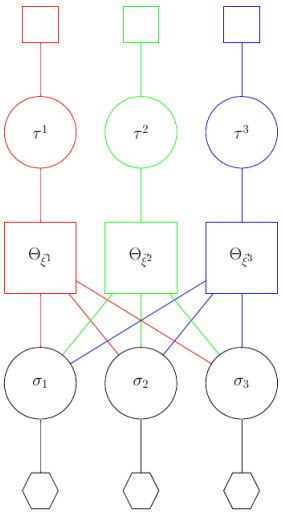

Figura 5.13: Esempio di grafo fattoriale con repliche associato ad un percettrone con $N = 3$ per tre pattern distinti dal colore.

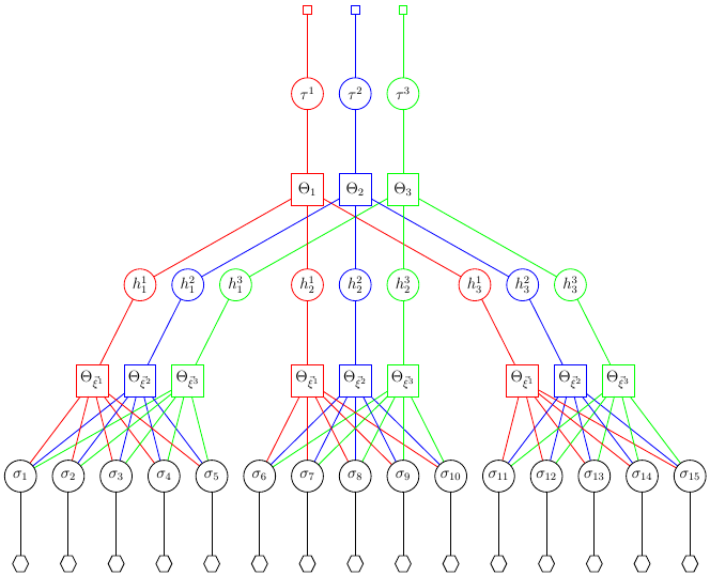

Figura 5.14: Esempio di grafo fattoriale con repliche associato ad una committee machine con $N = 15$ e $K = 3$ e con le rappresentazioni interne per tre pattern distinti dal colore.

5.3.2 Messaggi extra

Per quanto riguarda invece le modifiche analitiche dei messaggi, si può dimostrare (appendice F) che ogni nodo variabile σ_i riceve il seguente termine extra $m_{\star\to\sigma_i}$ dalle restanti $y-1$ repliche:

$$\hat{m}^{t+1}_{\star\to\sigma_i} = tanh\Big[(y-1)artanh(m^t_{\sigma_i\to\star}tanh\gamma)\Big]tanh\gamma, \tag{5.34}$$

dove $m^t_{\sigma_i\to\star}$ indica il messaggio trasmesso, allo step t, dal nodo variabile σ_i al nodo fattore γ contenente i contributi del resto del grafo fattoriale. A questo punto si riprenda l'analisi dei messaggi per il percettrone esposta nella sezione 5.10 e si inserisca il contributo 5.34.

La prima equazione in 5.5, che rappresenta il messaggio inviato dal nodo variabile del peso σ_i al nodo fattore μ, diventa:

$$m^t_{\sigma_i\to\mu} = tanh\left[\sum_{k\in\partial\sigma_i\setminus\mu} artanh(\hat{m}^t_{k\to\sigma_i}) + \hat{m}^t_{\star\to\sigma_i}\right], \tag{5.35}$$

mentre i messaggi dal nodo fattore μ ai nodi variabile e dall'output τ^μ a μ rimangono invariati.

Riassumendo, per il percettrone binario, la regola di aggiornamento della Belief Propagation in seguito all'integrazione del RE è eseguita sul messaggio per il termine extra:

$$\hat{m}^{t+1}_{\star\to\sigma_i} = tanh\Big[(y-1)artanh(m^t_{\sigma_i\to\star}tanh\gamma)\Big]tanh\gamma\ , \tag{5.36}$$

sui messaggi che partono dai nodi variabile associati ai pesi σ_i:

$$\begin{aligned} m^t_{\sigma_i\to\mu} &= tanh\Bigg[\sum_{k\in\partial\sigma_i\setminus\mu} artanh(\hat{m}^t_{k\to\sigma_i}) + \hat{m}^t_{\star\to\sigma_i}\Bigg] \\ m^t_{\sigma_i\to\star} &= tanh\Bigg[\sum_{k\in\partial\sigma_i} artanh(\hat{m}^t_{k\to\sigma_i})\Bigg] \ , \end{aligned} \tag{5.37}$$

sul messaggio inviato dal termine in output:

$$m^t_{\tau^\mu\to\mu} = tanh\Bigg[\sum_{k\in\partial\tau^\mu\setminus\mu} artanh(\hat{m}^t_{k\to\sigma_i})\Bigg] \tag{5.38}$$

e sui messaggi spediti dal nodo fattore μ:

$$\begin{aligned} \hat{m}^t_{\mu\to\sigma_i} &= \frac{\sum_{\sigma_i}\sigma_i\,\hat{\nu}^t_{\mu\to\sigma_i}(\sigma_i)}{\sum_{\sigma_i}\hat{\nu}^t_{\mu\to\sigma_i}(\sigma_i)} \\ \hat{m}^t_{\mu\to\tau^\mu} &= \frac{\sum_{\tau^\mu}\tau^\mu\,\hat{\nu}^t_{\mu\to\tau^\mu}(\tau^\mu)}{\sum_{\tau^\mu}\hat{\nu}^t_{\mu\to\tau^\mu}(\tau^\mu)} \ , \end{aligned} \tag{5.39}$$

avendo

$$\begin{aligned} \hat{\nu}^{t+1}_{\mu\to\sigma_i}(\sigma_i) &= \sum_{\tau^\mu,\{\sigma_j\}_{j\neq i}}\Big[\Theta\Big(\tau^\mu\Big(\sum_{j\in n(\mu)}\sigma_j\,\xi^\mu_j\Big)\Big)\Big(\frac{1+\tau^\mu m^t_{\tau^\mu\to\mu}}{2}\Big)\prod_{j\in n(\mu)\setminus i}\Big(\frac{1+\sigma_j m^t_{\sigma_j\to\mu}}{2}\Big)\Big] \\ \hat{\nu}^{t+1}_{\mu\to\tau^\mu}(\tau^\mu) &= \sum_{\{\sigma_j\}}\Big[\Theta\Big(\tau^\mu\Big(\sum_{j\in n(\mu)}\sigma_j\,\xi^\mu_j\Big)\Big)\prod_{j\in n(\mu)}\Big(\frac{1+\sigma_j m^t_{\sigma_j\to\mu}}{2}\Big)\Big]. \end{aligned} \tag{5.40}$$

Invece, come detto in sezione 5.1.2, l'aggiornamento per le CM è svolto adattando le equazioni del percettrone binario ad ogni sottografo a percettrone contenuto nel grafo fattoriale. Una volta raggiunta la convergenza della BP si ha:

$$m_i = tanh\Bigg[\sum_{k\in\partial\sigma_i} artanh(\hat{m}_{k\to\sigma_i}) + \hat{m}_{\star\to\sigma_i}\Bigg] \tag{5.41}$$

e la configurazione di pesi $\vec{\sigma}$ si ricava $\forall i$ da $\sigma_i = sgn(m_i)$.

5.4 Algoritmo rfBP

L'algoritmo descritto finora su NNs binarie è stato creato con i seguenti passaggi:

- sviluppo della distribuzione ripesata e non-vincolata (equazione 4.34) mediante il Robust Ensemble (equazione 5.29);

- rappresentazione della distribuzione attraverso i grafi fattoriali;
- dato $\beta \to \infty$ e fissati y e γ, esecuzione della BP considerando un'ipotesi euristica per le repliche e aggiungendo dei termini extra ai messaggi;
- dopo aver raggiunto la convergenza, determinazione dei pesi dal segno delle magnetizzazioni totali.

Tali passaggi definiscono l'algoritmo *replicated Belief Propagation*, che è la base dell'algoritmo rfBP. Quest'ultimo si ottiene modificando progressivamente almeno uno tra i parametri y e γ. Dato che l'aumento di y comporta l'individuazione di configurazioni con densità di entropia locale sempre più elevata e l'aumento di γ significa privilegiare le configurazioni con soluzioni progressivamente più vicine, il protocollo ideale sarebbe incrementare mano a mano entrambi i parametri. Nel caso di valori grandi per entrambi parametri, il sistema descrive configurazioni con alta densità d'entropia locale e a distanza molto piccola dalle soluzioni. La ricchezza di questi intorni e la prossimità alle soluzioni suggerisce che con alta probabilità le configurazioni raggiunte dall'algoritmo sono anch'esse soluzioni.

Di conseguenza, una volta integrato il RE nel grafo fattoriale e dato $\beta \to \infty$, l'algoritmo di rfBP può essere riassunto generalmente in:

- definire dei valori iniziali per y e γ;
- impostare un protocollo di crescita per tali parametri;
- tenendo fissi i parametri, eseguire la BP fino a convergenza;
- una volta raggiunta la convergenza, aumentare y e γ in base al protocollo fissato;
- rieseguire la BP fino a convergenza mantenendo fissi questi nuovi parametri;
- continuare a modificare i parametri ed eseguire la BP fino a che l'intero protocollo non è completato;
- al termine del protocollo, determinare i pesi dal segno delle magnetizzazioni totali.

Questo tipo di procedimento focalizza progressivamente il sistema a concentrarsi sulle configurazioni con intorni ricchi di soluzioni e molto vicini ad esse.

In letteratura, sono stati creati alcuni protocolli di aggiornamento dei parametri, alcuni che modificano solo uno tra i parametri e alcuni che li modificano entrambi. Uno dei più usati si basa sulla crescita di un parametro $\rho \in [0, 1]$ e sulle funzioni:

$$\begin{aligned} y &= \frac{2-\rho}{1-\rho} \\ \gamma &= artanh(\sqrt{\rho}). \end{aligned} \tag{5.42}$$

Per quanto riguarda la complessità computazionale dell'algoritmo rfBP, si sottolinea che il calcolo esatto dei messaggi della BP si può ottenere con complessità $O(N^3)$; mentre, con opportune approssimazioni, si può raggiungere una complessità pari a $O(N)$.

5.4.1 rfBP vs algoritmi euristici

L'algoritmo di rfBP può essere confrontato con il primo algoritmo euristico efficace creato, chiamato *reinforced Belief Propagation* (rBP) [38]. Quest'ultimo algoritmo si è rivelato efficace aggiungendo alle equazioni di BP un termine di rafforzamento che vincolasse i pesi ad assumere progressivamente un valore tra ± 1. Il termine extra aggiunto dal RE alle equazioni di BP può quindi essere ritenuto una quantità molto simile al termine rafforzativo dell'algoritmo di rBP.

Parziali sostegni a questa ipotesi sono dati sia dalle performance esibite dall'algoritmo rfBP che da un confronto analitico tra le equazioni dei messaggi di quest'ultimo e le equazioni dell'algoritmo rBP. Infatti, per il problema di classificazione sul percettrone binario l'algoritmo rfBP mostra una capacità massima $\alpha_C \simeq 0.6$, che è un valore inferiore ma non lontano dalla capacità massima dell'algoritmo rBP ($\alpha_C \simeq 0.7$). Inoltre, dal confronto analitico si può provare che per opportuni valori dei parametri y e γ le equazioni dei messaggi dell'algoritmo di rfBP coincidono con quelle dell'algoritmo rBP.

Questi fatti suggeriscono che l'utilizzo di distribuzioni fuori dall'equilibrio possa effettivamente dare accesso alle soluzioni degli algoritmi euristici efficaci. In altri termini, le capacità degli algoritmi euristici efficaci possono essere motivate parzialmente attraverso l'uso di distribuzioni al non-equilibrio definite in base alle proprietà delle soluzioni da essi ritrovate.

5.4.2 rfBP e densità di entropia locale

Le grandezze coinvolte nell'algoritmo rfBP possono essere anche utilizzate per calcolare la densità di entropia locale delle soluzioni. Questa possibilità permette il confronto con le osservazioni e i risultati analitici attesi dalla distribuzione ripesata e vincolata P_{RC} e dalla distribuzione ripesata e non vincolata P_{RU} (sezione 4.3.3).

Nella figura 5.15 è mostrato l'andamento della densità d'entropia locale media di un sistema vincolato ad una configurazione raggiunta mediante l'algoritmo rfBP (curva blu). Inoltre, sono presenti anche gli andamenti teorici della densità d'entropia locale per la P_{RC} nel caso $y = y^\star$ (curva rossa) e per la P_{RU} per $y \to \infty$ (curva verde). Come detto nel capitolo precedente, la densità di entropia locale di quest'ultimi casi sono quantitavamente molto simili. Nonostante ciò, dalla figura si può osservare che l'algoritmo rfBP è solidale con l'ipotesi di RS per le distanze non troppo elevate, oltre le quali diventa consistente con l'ipotesi di RSB. Ciò suggerisce che l'algoritmo di fBP sia in grado di gestire spontaneamente una rottura di simmetria.

Questo fatto può essere un altro indizio sul motivo per cui la capacità massima dell'algoritmo di rfBP è leggermente inferiore a quella degli algoritmi euristici. Una spiegazione ipotetica a ciò si può delineare nei seguenti termini:

- a distanze sufficientemente alte esiste un unico cluster denso di soluzioni dove l'ipotesi di RS vale;
- a distanze basse tale cluster si spezza in molteplici clusters altamente densi di soluzioni;
- la rfBP è in grado di cadere in uno di questi clusters;
- per capacità $\alpha > 0.6$, la frammentazione dell'unico cluster avviene attraverso rotture di simmetria di ordini elevati e l'algoritmo di rfBP perde la capacità di cogliere la nuova struttura dello spazio.

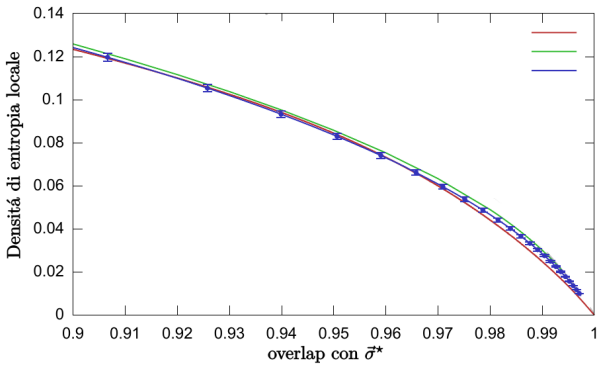

Figura 5.15: Densità locale di entropia (con α=0.6) vs overlap con la configurazione di riferimento $\vec{\sigma}^\star$. La curva in rosso rappresenta la densità di entropia locale per la P_{RC} calcolata con l'ipotesi di RS e $y^\star$ (sezione 4.3.1). La curva in verde indica la densità di entropia locale per P_{RU} ottenuta con l'ipotesi di RSB e $y \to \infty$ (sezione 4.3.2). La curva in blu (accompagnata dai punti) mostra invece l'andamento della densità di entropia locale ricava con l'algoritmo fBP (per y=21).

Capitolo 6

Applicazione di rfBP su dati di Genome Wide Association

Questo capitolo mostra un'applicazione dell'algoritmo rfBP a dati di Genome Wide Association (GWA) nel contesto della *Source Attribution*.

La prima sezione introduce l'ambito biologico dell'applicazione e descrive le caratteristiche dei dati analizzati.

La seconda sezione delinea i risultati dell'algoritmo rfBP sia in termini di performance individuali che di performance confrontate con alcuni dei più comuni classificatori nel panorama del Machine Learning. Inoltre, è presente anche un'analisi del tipo di informazioni traibili sui dati dall'algoritmo rfBP, confrontato ad un'analisi statistica standard basata sui test di associazione.

Infine, nella terza sezione sono riportate le idee per gli sviluppi futuri e le conclusioni sulle capacità dell'algoritmo rfBP nel trattare dati di GWA nell'ambito della Source Attribution.

6.1 Introduzione all'applicazione

Il dominio delle applicazioni dell'algoritmo rfBP non è facilmente individuabile. La difficoltà di individuazione è riconducibile a due aspetti: la nuova strategia di apprendimento e l'implementazione dell'algoritmo. Riguardo al primo aspetto, l'algoritmo rfBP è disegnato con l'obiettivo di raggiungere configurazioni che siano prossime a quelle configurazioni che memorizzano completamente il training set. Questo tipo di procedura è diversa dalle strategie tradizionali del Machine Learning, in cui si cerca tipicamente di evitare le configurazioni che memorizzano il training set, così da non imbattersi in problemi di *overfitting* [1] . Seppur questo tipo di strategia sembri dimostrare, da un punto di vista

[1]Si noti che per *overfitting* si intende l'effetto dovuto ad un'elevata aderenza con il training set e da una contemporanea debole capacità di generalizzazione.

teorico, un'ottima capacità di generalizzazione (figura 4.3), ciò non è ancora evidente su dati reali. La mancanza di un riscontro su dati reali frena l'uso comune di questa nuova strategia e favorisce gli algoritmi standard basati sulla discesa del gradiente.

L'altro aspetto limitante è l'implementazione dell'algoritmo rfBP. Fino ad oggi infatti, l'algoritmo rfBP è implementato solo per due strutture di Neural Networks: il percettrone e la committee machine. In aggiunta, l'implementazione coinvolge solo variabili di Ising e quindi le componenti dei vettori di input, i pesi e i valori in output possono essere solo ± 1. Siccome la maggior parte dei dati analizzati nel panorama del Machine Learning hanno natura continua e l'uso di NNs con numerosi hidden layer è sempre più frequente, un'implementazione di questo tipo restringe significativamente i campi di applicazione. Inoltre, l'implementazione originale dell'algoritmo è stata realizzata nel linguaggio di programmazione *Julia*, che nonostante stia nettamente aumentando di popolarità, non è ancora il linguaggio più usato per lo sviluppo di algoritmi per il Machine Learning.

Questo capitolo ha l'obiettivo di dimostrare che l'algoritmo rfBP, a prescindere dalle sue limitazioni pratiche, è un buon classificatore se applicato a dati genomici per la risoluzione di un problema di *Source Attribution*.

6.1.1 Genome Wide Association

In questa sezione sono introdotte alcune definizioni e nozioni biologiche necessarie a comprendere il contesto dell'applicazione.

Dati un set di genomi e una serie di classi, la *Genome Wide Association* mira ad assegnare un genoma ad una classe in base alle componenti che lo costituiscono. Il genoma è l'insieme dei geni di un organismo vivente, dove per geni si intendono le sequenze di nucleotidi del DNA utilizzate per la sintesi delle proteine [39]. I nucleotidi sono le unità del DNA e consistono in molecole formate da un gruppo fosfato, una molecola di zucchero e una base azotata. Nel DNA esistono quattro basi azotate: adenina (A), guanina (G), citosina (C) e timina (T).

Il DNA è formato da due polimeri di nucleotidi antiparalleli uniti dalla combinazione delle basi azotate mediante legami a idrogeno. Mentre le basi azotate assemblano i due filamenti antiparalleli, la catena costituita dai gruppi fosfato e dalle molecole di zucchero definiscono la struttura di supporto del DNA, chiamato *backbone*. Le informazioni più importanti del DNA sono quindi contenute nelle coppie di basi azotate, indicate con *bp* (base pair). Le coppie di basi azotate che caratterizzano il DNA sono adenina con timina (AT) e citosina con guanina (CG). In particolare, le basi azotate si possono distinguere in *purine*, adenina e guanina, e *pirimidine*, citosina e timina.

Il DNA di un individuo è quindi tipicamente rappresentato dalla sequenza di basi azotate che lo costituiscono. Come anticipato, esistono parti di questa catena, i geni, che sono trascritte e usate per la sintesi delle proteine. L'insieme dei geni forma il genoma.

Sequenziando i genomi [2] di tanti individui di una stessa specie, è possibile determinare il cosiddetto *genoma di riferimento*. Il genoma di riferimento di una specie sintetizza le caratteristiche principali dei geni dei singoli individui. Il confronto fra il genoma di riferimento di una specie con i genomi dei singoli individui permette di mettere in luce le variazioni che caratterizzano ognuno di tali individui. Queste variazioni possono essere condivise in tutte le cellule dell'organismo (*germinali*) o possono presentarsi solo al livello di singola cellula o di tessuti (*somatiche*). In particolare, vi è grande interesse riguardo alle variazioni su singola base che accadono in una specie con elevata frequenza. Questi tipi di variazioni sono dette SNPs, *Single Nucleotide Polymorphisms*, e sostanzialmente si realizzano quando una base nel genoma di riferimento è sostituita da un'altra base. Il genoma di un individuo è quindi rappresentabile anche in termini di SNPs ed in questo caso la GWA mira ad assegnare un genoma ad una classe in base ai suoi SNPs.

6.1.2 SNPs per Source Attribution

L'applicazione esposta in questo lavoro fa parte del progetto COMPARE finanziato dall'Unione Europea all'interno del programma di ricerca e innovazione Horizon 2020 (*grant agreement* No. 643476). Il progetto COMPARE ha come scopo lo sviluppo di metodologie e l'acquisizione di informazioni che permettano di evitare la trasmissione di malattie attraverso il cibo e di prevenire la nascita di nuove patologie infettive. In questo progetto è di particolare interesse la Genome Wide Association nell'ambito della *Source Attribution*. La Source Attribution consiste in generale nel riconoscere quale essere vivente abbia ospitato un dato batterio.

In questo lavoro sono stati presi in esame 210 campioni di Salmonella enterica, il batterio a cui è frequentemente dovuta l'infezione *salmonellosi*, con lo scopo di determinare l'animale da cui essi sono stati ospitati. L'obiettivo di questo studio è capire come i genomi dei batteri siano influenzati dal proprio ospite animale e se è possibile evidenziare delle caratteristiche che permettano di risalire dalla sequenza di SNPs all'ospite. Riuscire a trovare queste caratteristiche permetterebbe di migliorare il controllo degli animali che apportano delle modifiche potenzialmente infettanti al genoma del batterio. Una conoscenza di questo tipo favorirebbe di conseguenza il controllo delle infezioni ed il commercio di cibi più salutari.

Ciascun campione di Salmonella è rappresentato dal proprio genoma di $4857450\,bp$ (bp = coppie di basi) in termini di SNPs. In ogni base è indicata la presenza di una SNP con 1 e l'assenza di una SNP con 0.

I 210 campioni provengono da 5 diversi ospiti animali, le cui numerosità sono elencate in tabella 6.1.

[2] Si noti che per sequenziamento si intende l'insieme delle tecniche di laboratorio e algoritmiche [40,41] che consentono di determinare la sequenza di basi di un individuo.

Ospite animale	Suino	Pollame da carne	Pollame da uova	Anatra	Bovino
Numero campioni	159	34	4	11	2

Tabella 6.1: Elenco del tipo di animali ospitanti la Salmonella enterica e la loro numerosità.

In modo da comprimere la lunghezza dei campioni si è scelto di rimuovere tutte le basi con frequenza di SNPs tra tutti i campioni uguale a 0. In questo modo le sequenze di SNPs hanno dimensione ridotta a 8189 *bp* ed è possibile visualizzarne le caratteristiche in figura 6.1.

Siccome l'implementazione dell'algoritmo rfBP considera solo quantità binarie, si è scelto di raggruppare gli ospiti animali non suini e nell'allenare l'algoritmo rfBP con lo scopo di distinguere tra ospite suino e ospite non suino. Da un punto di vista statistico lo stesso problema può essere analizzato mediante dei test di associazione sulle frequenze di mutazione, dove per mutazione in questo caso si intende l'insorgenza di SNP.

Si consideri una base tra le 8189 *bp* e si contino il numero di ospiti suini (a) e il numero il ospiti non suini (b) con SNP in tale base. Allo stesso modo si contino il numero di ospiti suini (c) e il numero di ospiti non suini (d) privi di SNP sulla stessa base. Una volta ottenuti questi valori è possibile formare la matrice di contingenza (esempio in tabella 6.2) ed eseguire il test del χ^2 .

	Ospite suino	Ospite non suino
Numero di ospiti con SNP	a	b
Numero di ospiti senza SNP	c	d

Tabella 6.2: Esempio di matrice di contingenza per il test di associazione su singola base.

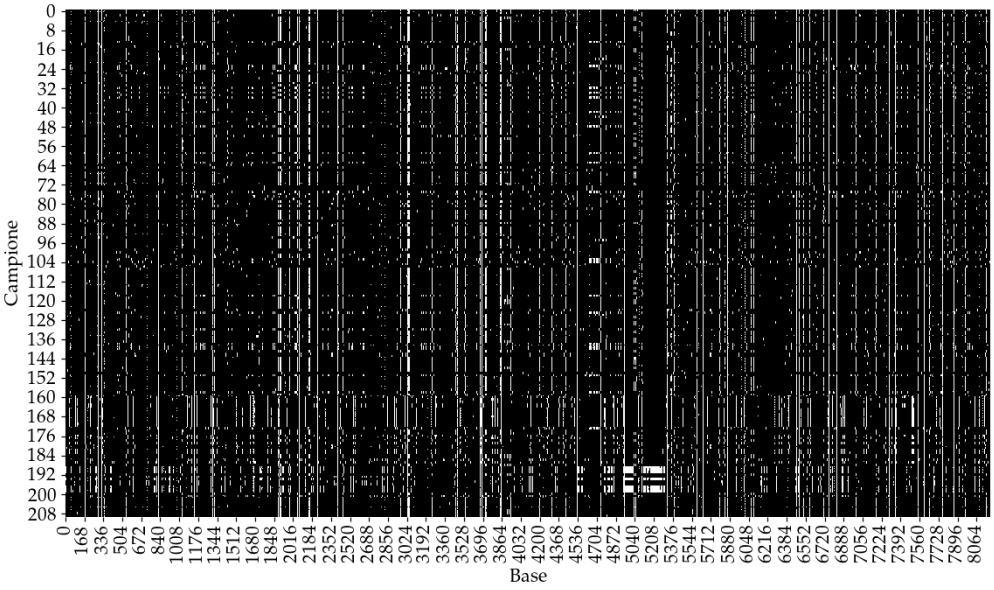

Figura 6.1: Sequenze di SNPs di tutti i campioni di Salmonella enterica. L'asse delle ascisse indicizza le basi nella sequenza, mentre l'asse delle ordinate indicizza i 210 campioni di Salmonella enterica. I punti neri e i punti bianchi identificano rispettivamente le basi prive di SNP e le basi con le SNP. Questa figura è stata organizzata in modo che le prime 159 righe contengano le sequenze dei batteri ospitati da suini e le restanti contengano le sequenze dei batteri ospitati dai non suini. Dalla figura è possibile distinguere qualitativamente il gruppo degli ospiti suini e una frazione del gruppo degli ospiti non suini.

Il test del χ^2 è stato eseguito per tutte le 8189 bp e quindi è stato necessario aggiustare i *p-values* ottenuti dai test singoli attraverso una correzione per test multipli. Utilizzando il metodo di correzione di Šidák [42] e definendo una soglia di significatività pari a 0.05, il numero di basi significative risulta uguale a 1103.

Un altro modo per analizzare i dati è verificare se il problema è risolvibile linearmente rispetto al numero di SNP dei singoli campioni. In questo caso, si conti per ogni campione il numero di SNPs presenti in tutta la sequenza. Ora, per vedere se è possibile trovare una soluzione lineare al problema rispetto a tali frequenze, sono state tracciate le distribuzioni del numero di SNPs per gli ospiti suini e per gli ospiti non suini sullo stesso grafico (figura 6.2).

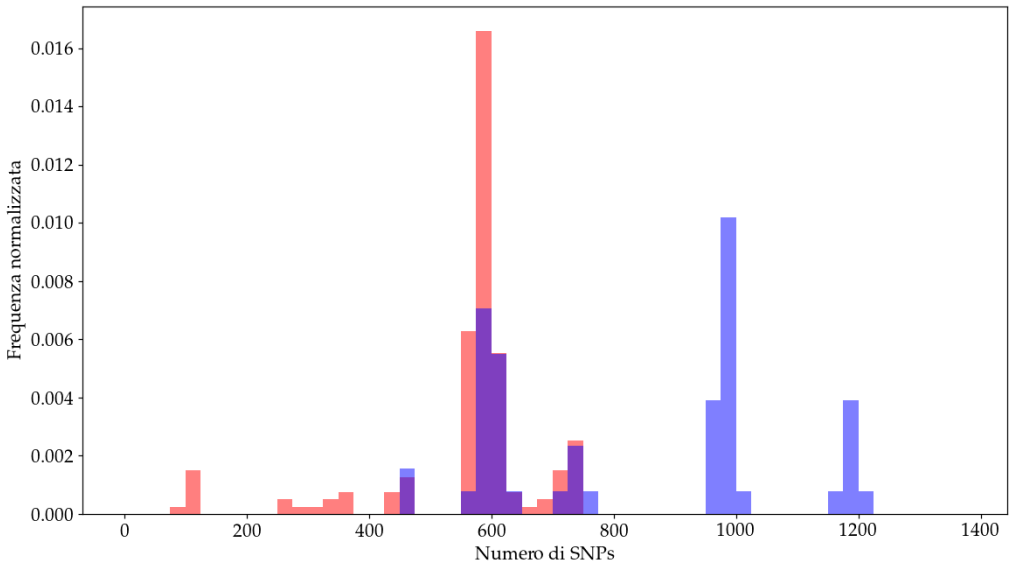

Figura 6.2: Distribuzione normalizzata del numero di SNPs per gli ospiti suini e per gli ospiti non suini. Si può osservare che esiste una parte di ospiti non suini che si può discriminare linearmente da tutti gli ospiti suini. Allo stesso tempo esiste una regione in cui le due distribuzioni sono sovrapposte, per cui l'informazione data dal solo numero di SNPs in una sequenza non è sufficiente per classificare linearmente tutti i campioni.

Come si può vedere dalla figura il problema non è risolvibile linearmente considerando solo il numero di SNPs dei campioni. Infatti, nonostante una frazione di ospiti non suini si possa discriminare linearmente dagli ospiti suini, è presente una regione in cui le distribuzioni di ospiti suini e non suini si sovrappongono. É quindi necessario sperimentare altri metodi per analizzare il problema, tra cui l'algoritmo rfBP, e confrontarli anche con il risultato dei test multipli del χ^2.

6.2 Analisi della rfBP

L'algoritmo rfBP è stato implementato in una nuova libreria C++, ottimizzata per il calcolo parallelo, ed è stato preparato un *wrapper* del codice per il porting su Python. Si è scelto di implementare l'algoritmo in tali linguaggi di programmazione per migliorare la velocità di esecuzione e per favorirne l'utilizzo. Infatti, il linguaggio C++ è tra i linguaggi standard nel settore della programmazione e permette di raggiungere prestazioni elevate in termini di velocità e gestione delle risorse. Il linguaggio Python è invece uno dei

linguaggi di programmazione più popolari del mondo ed è il linguaggio di riferimento per la comunità di Machine Learning.

Le sequenze di SNPs in esame sono state inizialmente processate trasformando gli 0 in -1, dato che l'algoritmo rfBP può trattare solo variabili di Ising (± 1). Inoltre, si è etichettata la classe dei suini con $+1$ e la classe dei non suini con -1.

Prima di effettuare il training dell'algoritmo è stato necessario definirne gli iper-parametri (sezione 1.2.1). Alcuni iper-parametri indispensabili sono il tipo di protocollo per i parametri y e γ (sezione 5.4), il numero di coppie (y, γ), il valore di soglia che definisce la convergenza dei messaggi e il numero di unità nell'hidden layer. Altri iper-parametri dovuti all'implementazione sono invece l'accuratezza nel calcolo dei messaggi (che può essere esatta o approssimata), il numero massimo di aggiornamenti della Belief Propagation (detto *epoch*) e un termine di regolarizzazione per l'evoluzione dei messaggi.

Per comodità si è scelto di utilizzare l'algoritmo rfBP su una struttura a percettrone e quindi il numero di unità nell'hidden layer è stato fissato ad 1. Inoltre, per semplificare la complessità computazionale si è scelto di adottare il calcolo approssimato dei messaggi. Il resto degli iper-parametri sono stati determinati attraverso un procedimento di ottimizzazione.

Inizialmente i 210 campioni sono stati suddivisi in una parte dedicata al training ed in una parte dedicata alla validazione (*validation set*). In particolare, si sono mantenute per entrambe le parti le stesse proporzioni tra le classi (suino e non suino), in modo da rimanere consistenti con l'insieme di tutti campioni. Una suddivisione che mantenga le proporzioni fra classi è detta *stratificata*.

In seguito, è stata eseguita l'ottimizzazione degli iper-parametri sul training set valutando le performance di ogni set di iper-parametri mediante una *cross-validazione* stratificata composta da 10 fold. In pratica, dal training set sono state generate casualmente 10 coppie di training subset e test subset, tali per cui ogni training subset rispettasse le proporzioni tra le classi e per cui ogni campione appartenesse al test subset una volta soltanto. L'algoritmo rfBP è stato allenato su ciascun training subset per prevedere le classi di appartenenza dei campioni nel test subset. Una volta completato questo procedimento su tutte le fold, si sono confrontate le classi predette dall'algoritmo rfBP con quelle attese. Per far ciò è stata sfruttata una nuova libreria sviluppata in C++ denominata *scorer*, che è in grado di calcolare numerose misure statistiche a partire dalla matrice di confusione attraverso un'innovativa strategia gerarchica sui grafi.

La misura statistica su cui si è quindi scelto di ottimizzare gli iper-parametri è il *coefficiente di correlazione di Matthews* (MCC) [43]. Tale coefficiente, compreso tra $[-1, 1]$, rappresenta una misura bilanciata dell'accuratezza nel prevedere entrambe le classi [3] .

[3] Si noti che si è preferito il coefficiente di correlazione di Matthews alla semplice *accuracy* (numero di corrette classificazioni) poichè le classi suino e non suino hanno una ben diversa popolosità, il che rende la semplice accuracy inaffidabile.

Il procedimento di ottimizzazione degli iper-parametri è stato svolto in 20 step, in cui i primi 3 step sono stati eseguiti su iper-parametri estratti casualmente e i rimanenti step sono stati governati da un processo *Bayesiano* basato su distribuzioni Gaussiane multivariate [44].

Una volta completata l'ottimizzazione, sono stati determinati i migliori iper-parametri ed è stato allenato l'algoritmo rfBP su tutto il training set. I pesi ottenuti dall'allenamento sul training set con i migliori iper-parametri sono stati quindi utilizzati per la predizione sul validation set.

Le percentuali dei campioni nel validation set sono state cambiate per valutare come l'algoritmo di rfBP si comporti per dimensioni crescenti del training set. In particolare, sono state considerate le seguenti percentuali per il training set: 25%, 45%, 65% e 85%.

6.2.1 Performance rfBP

Le perfomance in termini di accuracy e di coefficiente di correlazione di Matthews (MCC) dell'algoritmo rfBP sul validation set in seguito all'allenamento con i migliori iper-parametri su tutto il training set sono mostrate in figura 6.3.

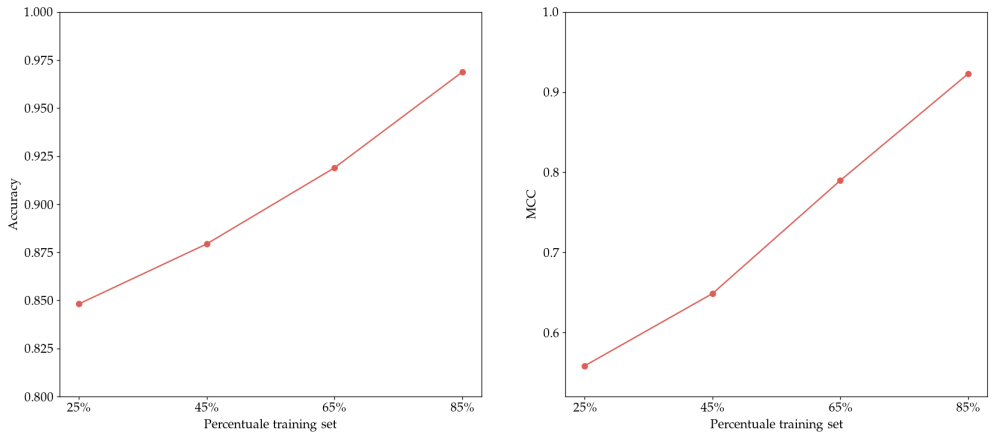

Figura 6.3: Accuracy e coefficiente di correlazione di Matthews (MCC) ottenuti dall'algoritmo rfBP sul validation set al crescere della dimensione del training set.

Dalle immagini si può osservare che l'aumento della dimensione del training set favorisce il miglioramento delle prestazioni dell'algoritmo rfBP sia per l'accuracy che per il coefficiente di Matthews.

Un altro aspetto interessante da valutare è il tipo di pesi, assegnati dall'algoritmo rfBP alle basi, che assieme ai migliori iper-parametri generano la migliore performance

sulla cross-validazione. Visto che le fold nella cross-validazione sono 10, si possono analizzare i 10 set di pesi ricavati rispettivamente dall'allenamento su ogni training subset. Come detto, tali 10 set di pesi (assieme ai migliori iper-parametri) ottengono la miglior performance nel cross-validare il training set ed è quindi lecito investigare le loro performance sul validation set. Si possono visualizzare le distribuzioni di performance di tali pesi al crescere della dimensione del training set in figura 6.4.

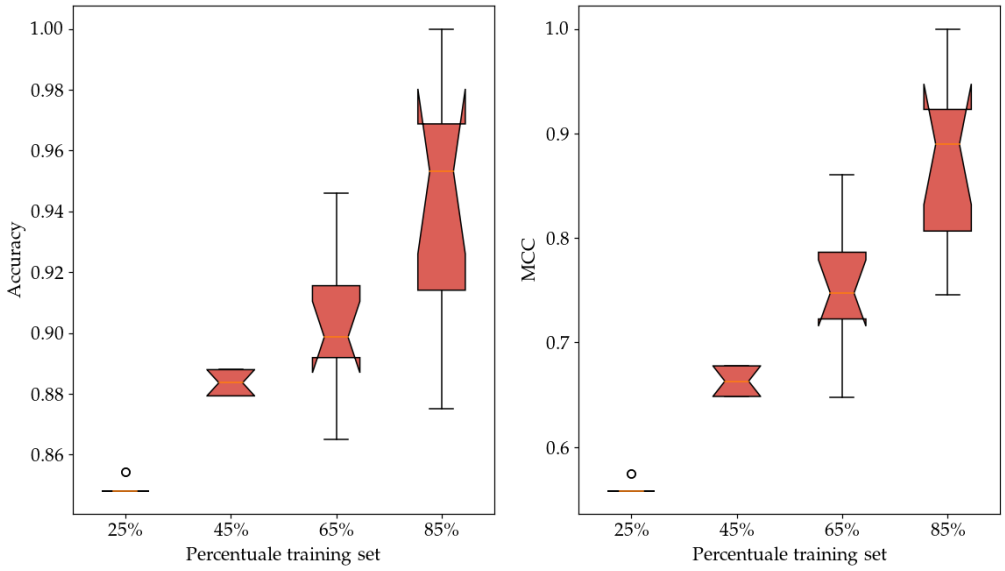

Figura 6.4: Distribuzione di accuracy e distribuzione del coefficiente di correlazione di Matthews (MCC) ottenuti dall'algoritmo rfBP sul validation set allenato sui 10 training subset, al crescere della dimensione del training set.

Queste performance sono ottenute dai singoli allenamenti sui 10 training subset. Siccome questi set sono sottogruppi del training set è normale aspettarsi un'oscillazione delle performance sul validation set attorno ai punti in figura 6.3. Ciò è infatti confermato osservando che le performance in figura 6.3 pressochè coincidano con i valori delle mediane dei boxplot in figura 6.4.

Dalle distribuzioni in figura 6.4 si può anche dedurre che l'allenamento su qualche training subset migliori la predizione sul validation set. Ad esempio, per una percentuale di training set pari a 85% l'allenamento dell'algoritmo rfBP su due training subset consente di raggiungere dei set di pesi in grado di classificare perfettamente il validation set (accuracy=1, MCC=1). Inoltre, in uno di tali casi, i pesi ottenuti sono anche in grado di classificare perfettamente il training subset. Queste elevate perfomance sono una prima

evidenza del fatto che l'algoritmo rfBP sia in grado di generalizzare ottimamente su dati reali, nonostante si avvicini ai pesi che memorizzano completamente il set di training.

6.2.2 Confronto con altri classificatori

L'algoritmo rfBP è stato confrontato con i seguenti classificatori [45–47]:

- Percettrone con pesi continui allenato mediante discesa del gradiente (indicato come Perc);
- Neural Network a multilayer allenato mediante discesa del gradiente (indicato come MLP);
- Support Vector Machine con kernel lineare (indicato come lSVM);
- Support Vector Machine con kernel radiale (indicato come rSVM) ;
- Linear Discriminant Analysis (indicato come LDA);
- Decision Tree (indicato come DT);
- Random Forest (indicato come RF);
- k-Nearest Neighbours fissando il numero di cluster a 2 (indicato come kNN);
- Gaussian Process (indicato come GP);
- Gaussian Naive Bayes (indicato come GNB);
- Bernoulli Naive Bayes (indicato come BNB);
- AdaBoost (indicato come AdaB).

Si è scelto di confrontare l'algoritmo rfBP per avere un'idea di quanto effettivamente tale algoritmo si discosti, in termini di performance, da alcuni dei più comuni classificatori. Per ogni dimensione del training set gli iper-parametri di tutti questi classificatori sono stati ottimizzati nello stesso numero di step dell'algoritmo rfBP. Le figure 6.5 e 6.6 mostrano il confronto delle performance ottenute sul validation set dai classificatori allenati su tutto il training set con i migliori iper-parametri.

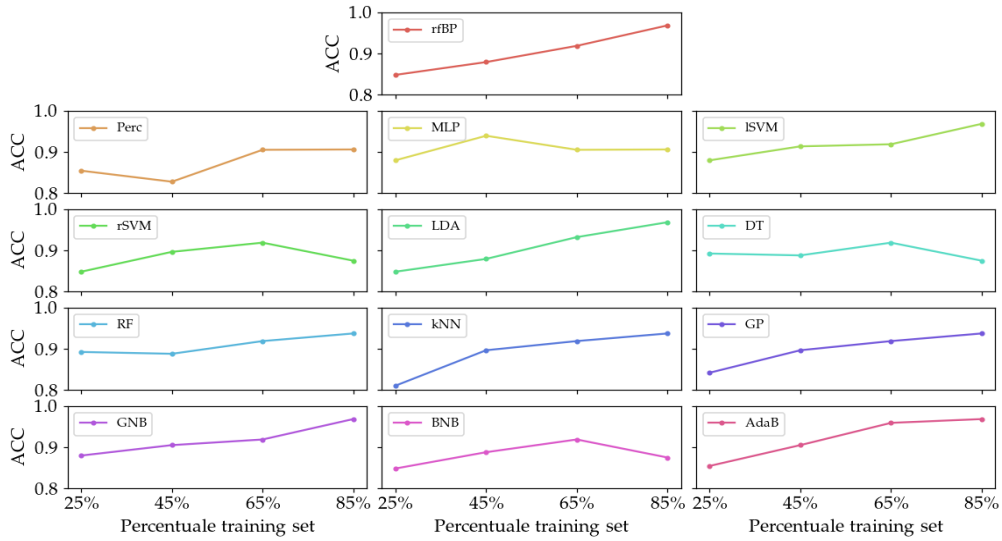

Figura 6.5: Confronto dell'accuracy sul validation set e dell'andamento al crescere della dimensione del training set tra l'algoritmo rfBP e alcuni classificatori standard.

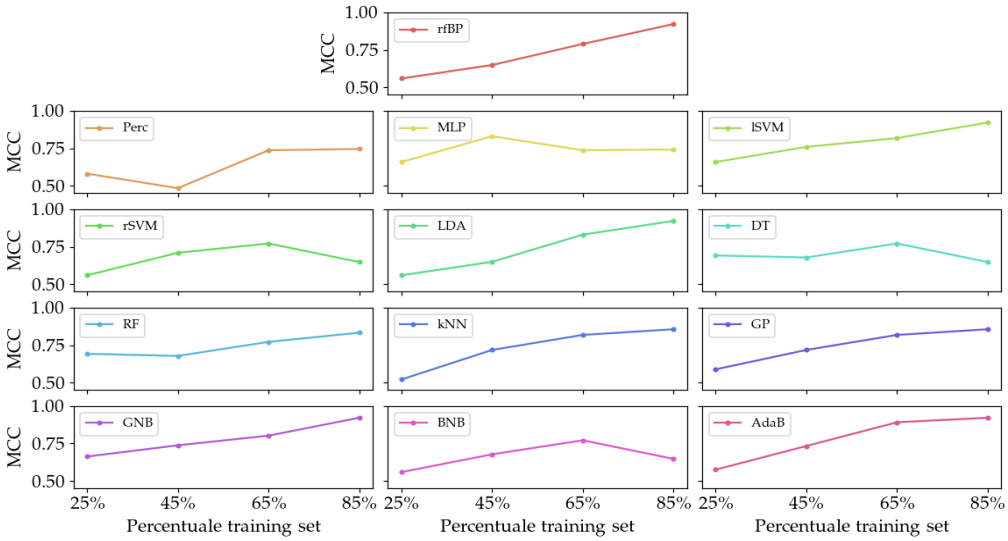

Figura 6.6: Confronto del coefficiente di correlazione di Matthews (MCC) sul validation set e dell'andamento al crescere della dimensione del training set tra l'algoritmo rfBP e alcuni classificatori standard.

É stato svolto un confronto analogo anche tra le distribuzioni delle performance sul validation set dei classificatori allenati sui 10 training subset (figure 6.7 e 6.8).

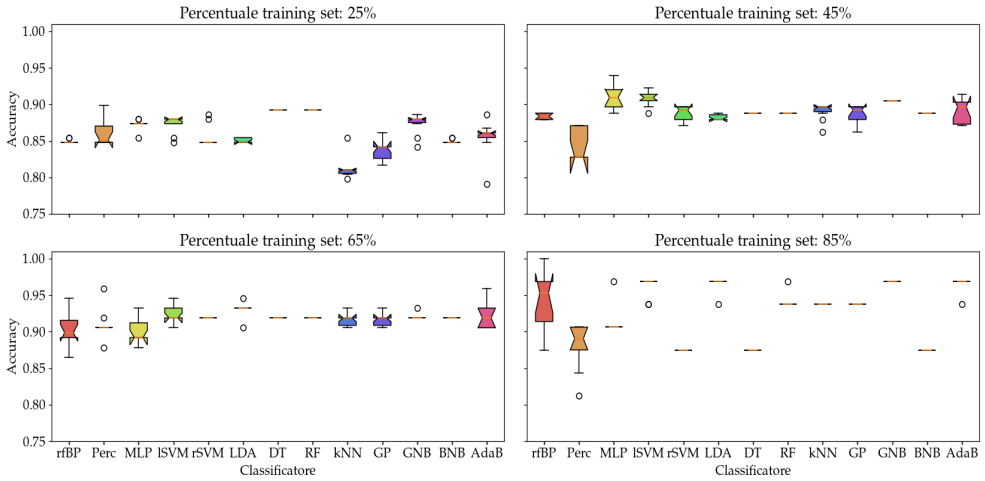

Figura 6.7: Confronto delle distribuzioni di accuracy sul validation set tra i classificatori allenati sui 10 training subset.

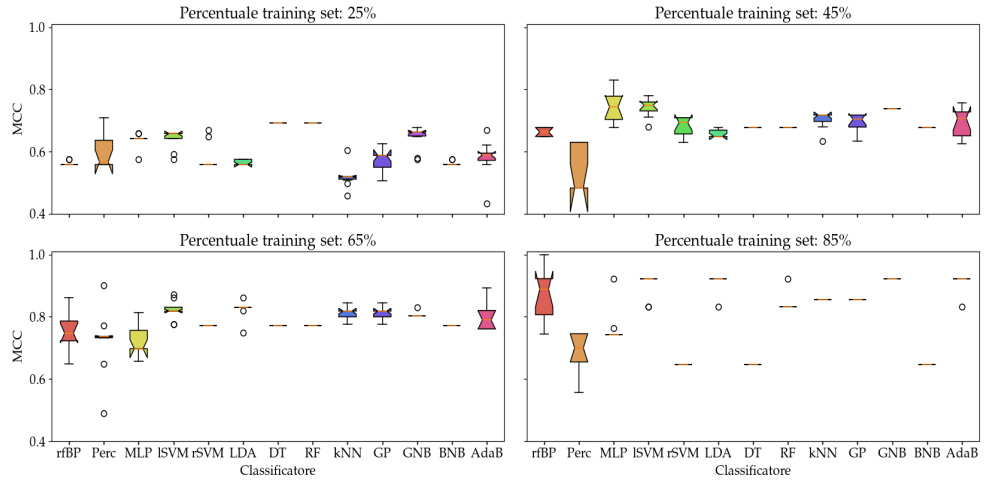

Figura 6.8: Confronto delle distribuzioni del coefficiente di Matthews sul validation set tra i classificatori allenati sui 10 training subset.

Dalle figure precedenti si può osservare che, in generale, l'algoritmo rfBP ha prestazioni in linea con tutti gli altri classificatori. Le prestazioni per tutti i classificatori migliorano all'aumentare del training set ma solo l'algoritmo rfBP riesce a raggiungere l'apprendimento perfetto sul validation test (accuracy=1 e MCC=1).

Dalle figure 6.5 e 6.6, si può inoltre vedere che l'algoritmo rfBP e il classificatore Gaussian Naive Bayes sono gli unici che qualitativamente non mostrano segni di saturazione delle performance. In aggiunta, il fatto che gli andamenti di accuracy e di MCC praticamente coincidano suggerisce che tutti i classificatori riescano a classificare proporzionalmente le classi di ospiti.

6.2.3 Confronto con i test multipli del χ^2

Infine, i risultati ottenuti dall'algoritmo rfBP sono stati analizzati rispetto alle informazioni date dai test multipli del χ^2. L'idea è verificare se l'algoritmo rfBP identifichi delle basi significative e se tali basi siano equivalenti alle basi significative dei test multipli del χ^2.

Questo tipo di confronto è stato condotto solo per la percentuale di training set uguale ad 85%, perchè con questa percentuale l'algoritmo rfBP realizza la miglior performance anche rispetto ai classificatori standard.

Per eseguire un confronto consistente tra i pesi dell'algoritmo rfBP e i risultati dei test multipli del χ^2 si è definito un metodo che traduca la significatività (o non significatività) di una base del genoma in un peso ideale con tre possibili valori: $\{+1, -1, 0\}$. Ad ogni base non significativa (p-value > 0.05) è stato assegnato peso ideale 0, mentre alle basi significative è stato assegnato $+1$ o -1. La scelta di assegnare $+1$ o -1 ad una base è convenzionale e quindi si è scelto di assegnare peso $+1$ ad una base se la classe dei suini è significativamente più mutata rispetto alla classe dei non suini, e -1 nel caso opposto. Si definisca il set di pesi generato con tale metodo *set di pesi ideali*.

In generale, siccome la classe dei suini e la classe dei non suini sono state rispettivamente etichettate con $+1$ e -1, la binarizzazione ideale dei pesi implica che tutte le basi significativamente mutate per i suini, $+1$, tendano ad associare il campione ai suini; mentre tutte le basi significativamente mutate per i non suini, -1, tendano invece ad assegnare il campione alla classe dei non suini. Questo meccanismo è in linea con il funzionamento di un percettrone, in cui ogni base è associata ad un peso sinaptico e la classe di un campione è determinata da una funzione di attivazione sulla combinazione lineare dei pesi e delle SNPs [4].

Usando questo metodo di conversione, solo le 1103 basi significative sono associate ad un peso ideale non nullo, come mostrato anche nella prima riga in figura 6.9. Si può anche osservare che la maggior parte di queste sono -1, ovvero che sono molto più

[4] Si ricordi che nel caso dell'algoritmo rfBP la funzione di attivazione è la Θ di Heaviside.

numerose le basi significativamente più mutate per i non-suini. In particolare, esistono 255 pesi ideali uguali a $+1$ e 889 pesi ideali pari a -1.

Riguardo ai pesi dell'algoritmo rfBP, si è deciso di non confrontare i pesi ottenuti dall'allenamento su tutto il training set ma di utilizzare i 10 set di pesi ottenuti dagli allenamenti sui training subset. Questa decisione è dovuta a due motivi principali. Il primo motivo è che considerando un singolo set di pesi dell'algoritmo rfBP non è possibile confrontare opportunamente i set di pesi ideali, visto che i primi sono caratterizzati da 2 possibili stati (± 1) e i secondi da 3 (± 1 e 0). Il secondo motivo è che da un singolo set di pesi non è possibile capire il criterio di discriminazione che l'algoritmo rfBP individua per classificare i campioni.

Di conseguenza si è optato per considerare i set di pesi ottenuti dall'allenamento sui training subset. Questi set di pesi sono i precursori del singolo set ricavato dall'allenamento su tutto il training set, visto che questi parametri si sono dimostrati, al termine dell'ottimizzazione degli iper-parametri, i migliori nel predire il training set. Come visto in precedenza, ciò è ulteriormente giustificato dal fatto che le performance di tali set di pesi sul validation set (figure 6.7 e 6.8) interpolino circa nella mediana le performance dei pesi ottenuti sull'intero training set (figure 6.5 e 6.6).

Da questi 10 set di pesi è stato calcolato un set di pesi rappresentativo delle informazioni trovate dall'algoritmo rfBP, che si definisce *set di pesi rfBP*. Il set di pesi rfBP è stato calcolato come la media puntuale dei 10 set di pesi e si può osservare all'ultima riga in 6.9. In questo modo, come si può dedurre anche dalla figura, ogni componente del set di pesi rfBP non è più vincolato ai valori ± 1.

Il set di pesi rfBP risulta contenere 5201 pesi uguali ad $+1$ o -1. Ciò significa che l'algoritmo rfBP assegna sempre lo stesso peso a 5201 basi per ogni allenamento sui 10 training subset. Si può quindi ipotizzare che l'algoritmo rfBP riesca a ricavare dalla sola combinazione di queste basi la componente discriminante principale tra le due classi. Tali 5201 possono quindi essere definite come le basi significative dell'algoritmo rfBP.

Il confronto tra il set di pesi ideali dei test multipli del χ^2 e il set di pesi rfBP mette in luce numerose differenze. La differenza più rilevante è l'incogruenza tra le basi significative attese dai test multipli del χ^2 e quelle significative dall'algoritmo rfBP. Si osservi la figura 6.10, che mostra le sequenze solo nelle basi significative dei test multipli del χ^2. Dal confronto tra la prima riga della figura e l'ultima si può notare che la maggior parte delle basi significative dei test multipli del χ^2 siano anche basi significative per l'algoritmo rfBP. In dettaglio si può trovare che:

- l'algoritmo rfBP trova esattamente la stessa significatività per 841 basi rispetto alle 1103 basi significative per i test multipli del χ^2;
- l'algoritmo rfBP individua correttamente 838 basi significative con segno -1 sulle 848 attese dai test multipli del χ^2, e 3 basi significative con segno $+1$ sulle 255 attese.

Da questi risultati si può concludere che l'algoritmo rfBP non riesca ad identificare quasi tutte le basi significativamente più mutate per i suini (+1). É possibile però trovare una giustificazione teorica alla perdita di tali basi. Si noti infatti che il set di pesi ideale è stato calcolato dai test multipli del χ^2 considerando tutti i campioni; mentre il set di pesi rfBP è stato calcolato sulla media dei set di pesi ottenuti dall'allenamento sui 10 training subset. In altre parole, il confronto non è del tutto consistente perchè il set di pesi rfBP non è stato ricavato considerando tutti i campioni.

Per svolgere un confronto consistente si è quindi scelto di calcolare un nuovo set di pesi usando i test multipli del χ^2. In primo luogo, si sono eseguiti su ciascun training subset i test multipli del χ^2. In questo modo si ricavano 10 set di pesi che rappresentano le basi significative trovate per ciascun training subset. In secondo luogo, effettuando una media puntuale di questi 10 set di pesi, si è creato un nuovo set di pesi, chiamato *set di pesi attesi*. Il set di pesi attesi, mostrato alla seconda riga in figura 6.9, è costituito da valori reali come il set di pesi rfBP. Si può osservare dalle figure 6.9 e 6.10 che il set di pesi attesi non ritrova tutte le basi significative date dal set di pesi ideali. In particolare:

- il set di pesi attesi trovano 702 basi significative sulle 1103 indicate dal set di pesi ideali;
- il set di pesi attesi individua correttamente 696 basi significative con segno -1 sulle 848 del set di pesi ideali, e 6 basi significative con segno $+1$ sulle 255 del set di pesi ideali.

Questi risultati suggeriscono che i training subset non contengano informazioni sufficienti per determinare una significatività delle basi associate al peso $+1$. Ciò può spiegare perchè l'algoritmo rfBP non identifichi tali basi come significative.

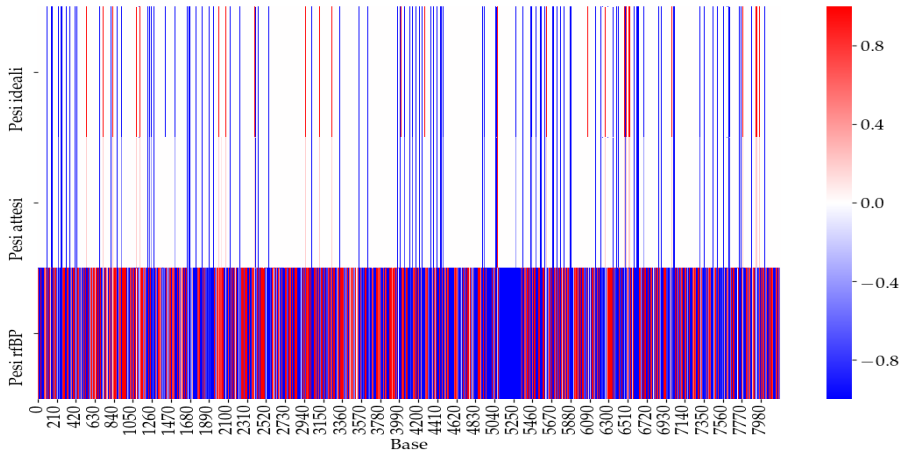

Figura 6.9: Confronto tra il set di pesi ideale, il set di pesi atteso e il set di pesi rfBP. Il colore indica il livello di significatività di una base: più un peso tende a $+1$ (rosso) o -1, più la base è significativa (blu). Una base con peso $+1$ è una base significativamente più mutata per i suini. Una base con peso -1 è una base significativamente più mutata per i non suini.

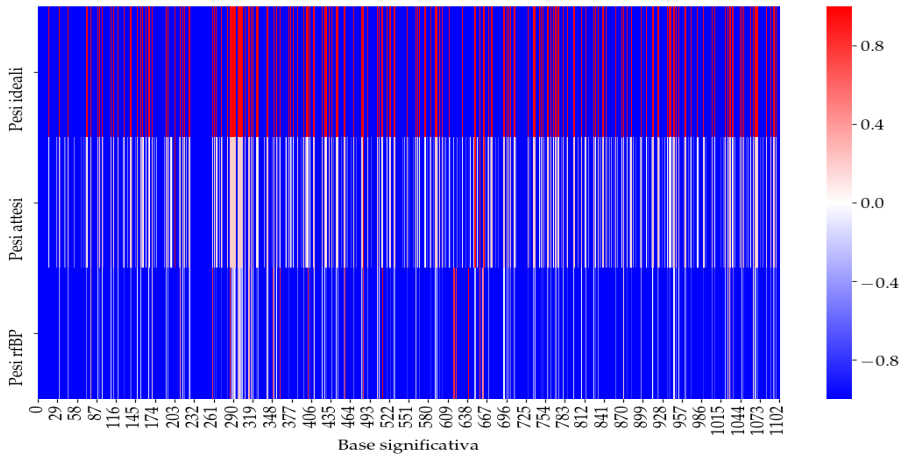

Figura 6.10: Confronto tra il set di pesi ideale, il set di pesi atteso e il set di pesi rfBP solo per le basi in cui il set di pesi ideale è non nullo. Il colore indica il livello di significatività di una base: più un peso tende a $+1$ (rosso) o -1, più la base è significativa (blu). Una base con peso $+1$ è una base significativamente più mutata per i suini. Una base con peso -1 è una base significativamente più mutata per i non suini.

A questo punto, due ipotesi possono essere investigate. La prima è che effettivamente le sole basi significative ottenute dai test multipli del χ^2 (ovvero le basi con peso ideale non nullo) siano sufficienti a discriminare le classi. La seconda è che l'algoritmo rfBP sia in grado invece di trovare una combinazione tra tali basi significative e le basi non significative che consenta di favorire la classificazione. Questa seconda ipotesi consiste quindi nell'utilizzare solo le basi associate a pesi $+1$ o -1 nel set di pesi rfBP.

Per verificare queste ipotesi sono stati rieseguiti i passaggi di ottimizzazione e allenamento, indicati in sezione 6.2.1, sugli stessi campioni, rappresentati prima dalle basi significative per i test multipli del χ^2 e poi dalle basi significative per l'algoritmo rfBP. Le figure seguenti 6.11 e 6.12 mostrano le performance, sia in termini di accuracy che in termini di coefficiente di Matthews.

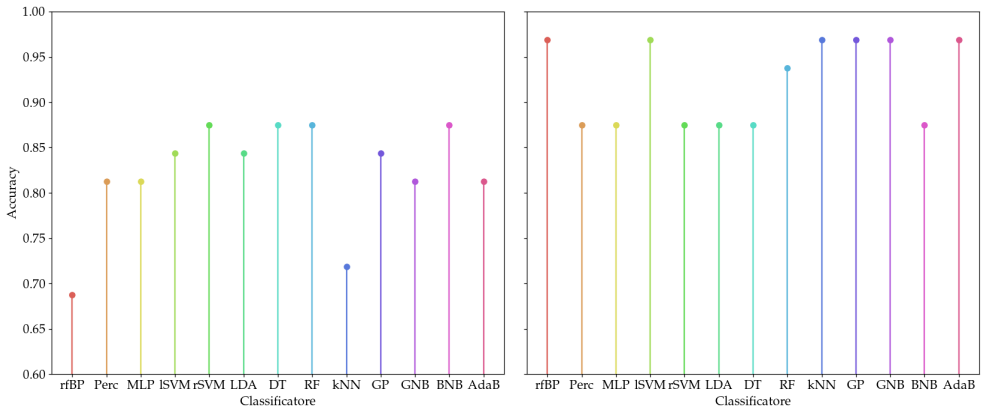

Figura 6.11: Confronto tra l'accuracy dei classificatori allenati solo sulle basi significative per i test multipli del χ^2 (a sinistra) e l'accuracy dei classificatori allenati solo sulle basi significative dell'algoritmo rfBP (a destra). É stato utilizzato l'85% dei campioni nel training set.

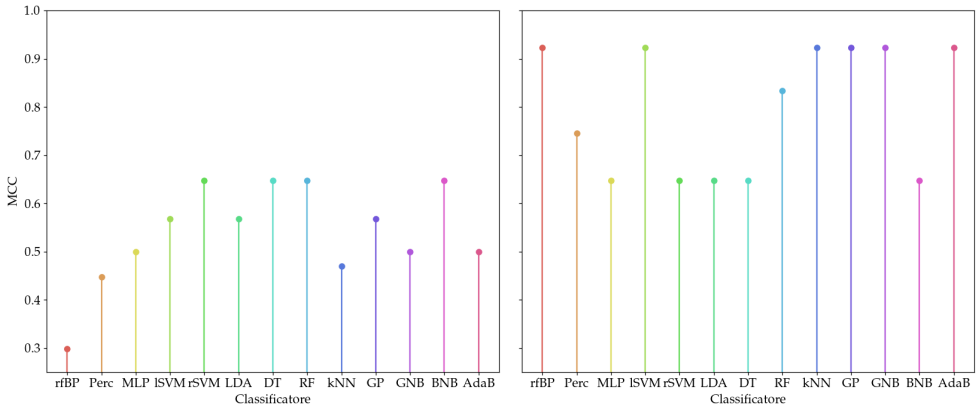

Figura 6.12: Confronto tra il coefficiente di Matthews (MCC) dei classificatori allenati solo sulle basi significative per i test multipli del χ^2 (a sinistra) e il MCC dei classificatori allenati solo sulle basi significative dell'algoritmo rfBP (a destra). É stato utilizzato l'85% dei campioni nel training set.

Le precedenti figure mostrano che l'uso delle basi significative dell'algoritmo rfBP è migliore, per ogni classificatore, dell'uso delle basi significative trovate dai test multipli del χ^2. Questo fatto implica che le basi significative dell'algoritmo rfBP contengano effettivamente più informazioni per la classificazione rispetto alle basi significative dei test multipli del χ^2.

A questo punto, è utile anche verificare quanto si discostano le performance dei classificatori allenati solo sulle basi significative dell'algoritmo rfBP da quelle ottenute considerando tutte le 8189 basi. Le figure seguenti 6.13 e 6.14 mostrano il confronto.

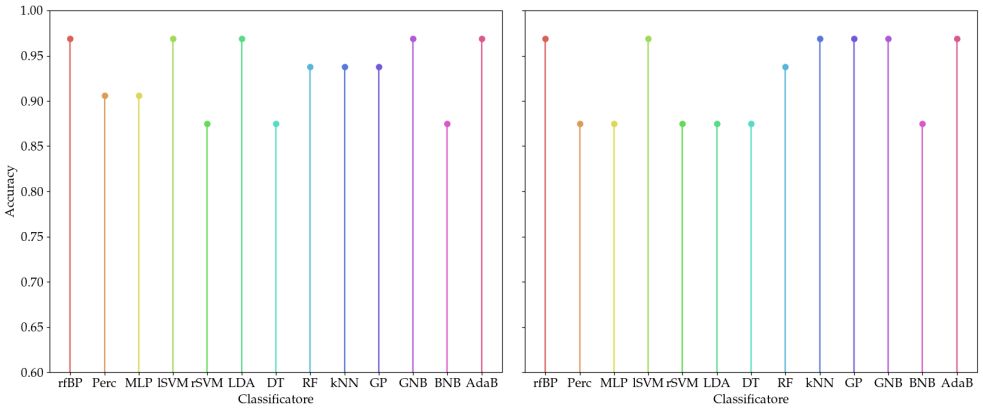

Figura 6.13: Confronto tra l'accuracy dei classificatori allenati su tutte le basi (a sinistra) e l'accuracy dei classificatori allenati solo sulle basi significative dell'algoritmo rfBP (a destra). É stato utilizzato l'85% dei campioni nel training set.

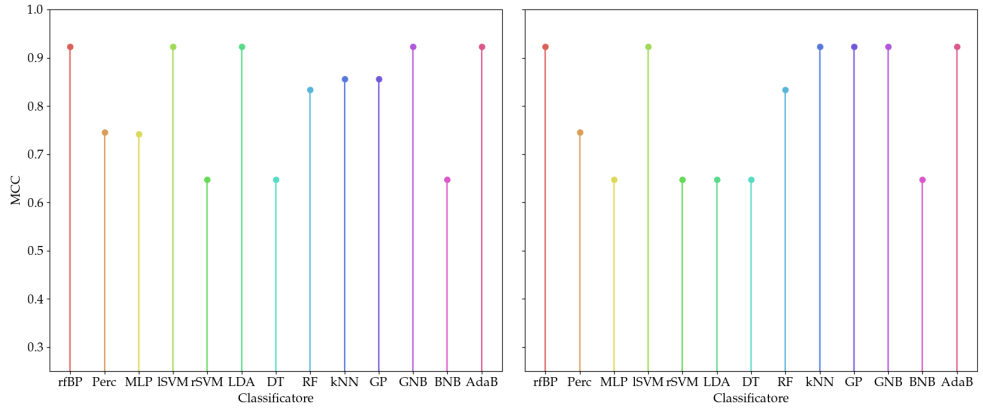

Figura 6.14: Confronto tra il coefficiente di Matthews (MCC) dei classificatori allenati su tutte le basi (a sinistra) e il MCC dei classificatori allenati solo sulle basi significative dell'algoritmo rfBP (a destra). É stato utilizzato l'85% dei campioni nel training set.

Dalle figure 6.13 e 6.14 si può osservare che l'algoritmo rfBP non modifica le sue performance, rafforzando l'ipotesi che siano sufficienti le sue 5201 basi significative per distinguere ottimamente le classi. I classificatori LDA e MLP sono gli unici che mostrano

un peggioramento delle performance [5] , mentre tutti gli altri o mantengono le stesse performance o le migliorano.

In conclusione, i risultati ottenuti supportano l'idea che l'algoritmo rfBP possa essere utilizzato efficacemente per l'analisi di dati genomici, come le SNPs, ed incoraggiano ad indagare ulteriormente il tipo di basi significative trovate da tale algoritmo.

[5] Si noti che questo peggioramento potrebbe anche essere dovuto a problematiche numeriche e di convergenza durante il training dei classificatori.

Capitolo 7

Conclusioni

Questo lavoro ha ripreso le teorie fisico-statistiche sul fenomeno dell'apprendimento e ha spiegato come l'integrazione di modelli a Spin-Glass, grafi e Neural Networks permettano di impostare lo sviluppo di algoritmi per il Machine Learning. Allo stesso tempo, questo lavoro si è focalizzato su un filone di ricerca di fisica che propone di utilizzare sistemi al non-equilibrio per migliorare le capacità di apprendimento di una Neural Network. Da un punto di vista fisico, la descrizione del fenomeno dell'apprendimento attraverso distribuzioni al non equilibrio si traduce nel definire dei processi che mirino a massimizzare una cosiddetta *densità di entropia locale* del sistema. Questi tipi di processi si discostano dai metodi standard di discesa del gradiente utilizzati nel settore del Machine Learning.

Questo lavoro ha avuto inoltre l'obiettivo di spiegare il funzionamento dell'algoritmo *replicated focusing Belief Propagation* (rfBP), che è disegnato su una distribuzione al non-equilibrio ed è sviluppabile come una versione parametrizzata e rafforzata dell'algoritmo Belief Propagation (BP).

Infine, questo lavoro ha presentato un'applicazione dell'algoritmo rfBP su dati di Genome Wide Association (GWA) nell'ambito della Source Attribution. L'uso pratico di algoritmi come l'algoritmo rfBP non è ancora popolare tanto quanto gli algoritmi standard sulla discesa del gradiente, sia per ragioni dovute alle implementazioni di tali algoritmi che per la mancanza di risultati che evidenzino come questi nuovi algoritmi possano essere usati efficacemente su dati reali.

Questo lavoro ha dimostrato che l'algoritmo rfBP è in grado di risolvere ottimamente la classificazione di sequenze di SNPs (*Single Nucleotide Polymorphism*). Le performance di questo algoritmo sono state confrontate con i più comuni classificatori utilizzati nel settore del Machine Learning.

Sono stati considerati campioni di genomi del batterio Salmonella, ospitati in diversi animali, ed è stato effettutato il training dell'algoritmo rfBP sull'insorgenza di SNPs nel tentativo di classificare l'ospite in base a tali polimorfismi. Questa applicazione ha fatto parte del progetto COMPARE finanziato dall'Unione Europea all'interno del programma di ricerca e innovazione Horizon 2020 (*grant agreement* No. 643476).

La prima conclusione che si può trarre dai risultati dell'applicazione è che le sequenze di SNPs dei campioni a disposizione possono essere analizzate ottimamente mediante l'utilizzo dell'algoritmo rfBP eseguito su una struttura a percettrone. Ciò implica che i dati analizzati possano essere discriminati linearmente con ottime performance e ciò è confermato dal fatto che le performance di classificatori lineari come il Linear Discriminant Analysis e la Support Vector Machine con kernel lineare siano anch'esse ottime.

Allo stesso tempo, però, il percettrone continuo si dimostra tra i peggiori classificatori indipendentemente dalla percentuale del training set. Ciò può essere spiegato ipotizzando che la discesa del gradiente non sia in grado di localizzare un criterio di classificazione ottimale. Questa ipotesi va a favore dello sviluppo di algoritmi come l'rfBP, che mirano a minimizzare o massimizzare quantità diverse dall'energia o da un'analoga funzione costo.

La seconda conclusione è che l'algoritmo rfBP dimostra di essere all'altezza dei più tradizionali algoritmi di Machine Learning nonostante tutte le quantità coinvolte siano vincolate ad essere quantità binarie. Ciò suggerisce in primo luogo che la rappresentazione binaria delle sequenze di SNPs contenga informazioni sufficienti per risalire all'ospite e, in secondo luogo, che il limite binario dell'algoritmo rfBP possa in realtà rivelarsi un vantaggio rispetto a tutti gli altri classificatori per questa tipologia di dati. In aggiunta, la crescita delle performance dell'algoritmo per dimensioni crescenti del training set può essere motivata da un punto di vista teorico. La strategia su cui si fonda l'algoritmo rfBP è raggiungere set di pesi che siano vicini a quei pesi che memorizzano completamente il training set. Di conseguenza, più è popoloso il training set, più è probabile che la maggior parte di variabilità sia compresa in tale set. Quindi, l'algoritmo rfBP nel tentativo di avvicinarsi ai set di pesi che memorizzano il training set, ovvero che spiegano tutta la variabilità, ne acquisisce le proprietà principali, che potenzialmente diminuiscono l'errore di generalizzazione.

La terza conclusione è che le basi significative per l'algoritmo rfBP siano di gran lunga superiori nell'incentivare le performance di tutti gli algoritmi, rispetto alle basi significative ottenibili mediante test multipli del χ^2. Ciò suggerisce che l'algoritmo rfBP possa essere utilizzato come filtro del rumore o che alternativamente possa essere utilizzato come strumento per la *features selection*.

La quarta conclusione è che l'uso della Belief Propagation, rafforzato dal termine extra che caratterizza le equazioni dell'algoritmo rfBP, permetta effettivamente di andare oltre l'informazione sulla singola base (come i test χ^2) e di valutare un comportamento organico delle basi. In altre parole, l'algoritmo rfBP raggiunge un set di pesi sfruttando un meccanismo di interazione tra i pesi, che non si focalizza su informazioni locali (singola base) ma su informazioni marginali (aggregazione di informazioni locali).

La quinta conclusione è che l'efficienza dell'uso dell'algoritmo rfBP per classificare sequenze di SNPs nell'ambito della Source Attribution, può essere ulteriormente incentivata da una possibile implementazione hardware. Infatti, a differenza dei classificatori a pesi continui, l'algoritmo rfBP, una volta allenato, genera un set di pesi binari che possono essere facilmente implementati su un dispositivo hardware per un utilizzo dedicato.

La possibilità di dotarsi di un dispositivo hardware dedicato è in linea con i moderni sviluppi tecnologici, che aspirano sempre più a fornire strumenti dedicati che possano garantire una diffusione delle più avanzate tecnologie mediante una facile integrabilità.

7.1 Sviluppi teorici futuri

I risultati teorici ritrovati dall'algoritmo rfBP incoraggiano a continuare ad approfondire le conoscenze relative agli algoritmi disegnati sulle distribuzioni al non-equilibrio per sviluppare nuovi futuri algoritmi capaci di gestire autonomamente le rotture di simmetria nello spazio delle soluzioni. Come osservato nella sezione 5.4.2, l'algoritmo rfBP è in grado di gestire la rottura di un unico cluster denso di soluzioni fino a capacità $\alpha < 0.6$. Siccome l'algoritmo mostra tale effetto anche grazie all'ipotesi euristica definita in sezione 5.3.1, un futuro compito è quello di giustificare la ragione dell'efficacia di tale ipotesi e come essa governi l'evoluzione dell'algoritmo nello spazio delle soluzioni.

É inoltre al centro degli sviluppi futuri lo studio approfondito di sistemi fuori dall'equilibrio caratterizzati da variabili non binarie, andando oltre il dominio delle variabili di Ising. In aggiunta, può essere potenzialmente determinante verificare che tipo di risultati si ottengano massimizzando la densità di entropia locale nel caso di variabili continue, in contrapposizione alle discese del gradiente standard.

Infine, come suggeriscono le ultime ricerche di questo ramo di fisica statistica, è necessario iniziare a prendere in considerazione l'integrazione di fenomeni quantistici [48], la risoluzione dei problemi di inferenza sui Big Data [49,50], il ruolo degli effetti stocastici nell'apprendimento [51] e lo sviluppo di metodi per l'*unsupervised learning* [52].

7.2 Sviluppi applicativi futuri

I risultati positivi mostrati dall'algoritmo rfBP sull'applicazione nell'ambito della Source Attribution incoraggiano ad approfondire i margini di applicazione di questo algoritmo e di tutti gli algoritmi sviluppati sugli stessi costrutti teorici.

Una prima idea di sviluppo è estendere l'analisi sulle sequenze di SNPs cercando una classificazione a multiclasse. Siccome l'algoritmo è limitato all'utilizzo di output binari, un possibile procedimento consiste nello scegliere progressivamente le classi più numerose e mirare a classificare attraverso la regola *one versus all*. In questo modo si può diramare un'albero di classificazione per la generalizzazione su più di una classe. Inoltre, per verificare se l'algoritmo rfBP sia intrinsecamente adatto ad analizzare sequenze di SNPs, un'altra idea è trovare un'altra applicazione sempre su sequenze di SNPs ma con uno scopo diverso dalla Source Attribution. Un ulteriore compito futuro è osservare come varino le performance dell'algoritmo su una committee machine con un numero variabile di unità hidden.

In generale, i settori su cui si ha intenzione di sviluppare altre applicazioni sono specialmente i settori biomedici tra cui: l'analisi di immagini biomediche (es. NMR, TAC, PET), di segnali biologici (es. Heart rate variability e Brain health monitoring) e problemi di genetica ed epigenetica (es. metilazione).

Infine, come anticipato in precedenza, è di estremo interesse verificare per ogni applicazione dell'algoritmo rfBP se sia potenzialmente conveniente sviluppare dispositivi hardware dedicati che integrino il set di pesi binari ottenuti dall'algoritmo.

Appendice A

Grafi fattoriali ed expression trees

Questo appendice ha lo scopo di chiarire la relazione tra i grafi fattoriali e gli expression trees, superando il parallelismo qualitativo suggerito nella sezione 2.2.1. La maggior parte dei passaggi è ripresa dall'appendice A dell'articolo in [7].

Sia $g(\vec{x})$, con $\vec{x} = (x, x_1, x_2, .., x_{n-1})$, una funzione reale fattorizzabile e descrivibile attraverso un grafo ad albero. Si consideri il marginale:

$$g(x) = \sum_{\sim x} g(\vec{x}), \tag{A.1}$$

e si fissi come riferimento del grafo la variabile x.

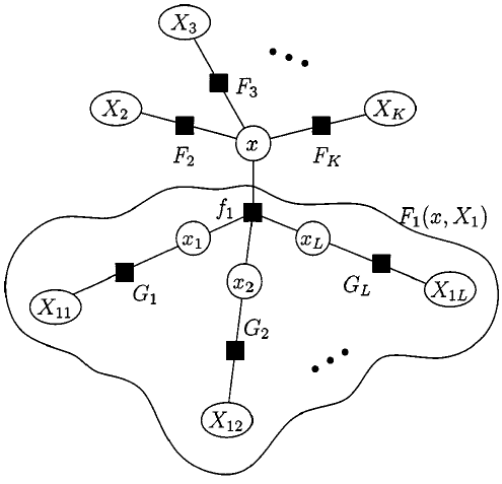

Figura A.1: Grafo fattoriale ad albero generale.

Ipotizzando che esistano K funzioni che dipendano da x, la funzione globale si può scrivere come:

$$g(\vec{x}) = \prod_{k=1}^{K} F_k(x, \vec{X}_k). \tag{A.2}$$

Visto che il grafo considerato è ad albero, il set dei $\vec{X}_k$ è una partizione di $\vec{x}$, ovvero $\vec{X}_k \cap \vec{X}_l = 0$ per $k \neq l$ e $\cup_k \vec{X}_k = (x_1, x_2, .., x_{n-1})$. Sfruttando la proprietà distributiva e il fatto che le $\vec{X}_k$ siano una partizione:

$$g(x) = \sum_{\sim x} \prod_{k=1}^{K} F_k(x, \vec{X}_k) = \prod_{k=1}^{K} \sum_{\sim x} F_k(x, \vec{X}_k). \tag{A.3}$$

Ora si analizzi la sommatoria $\sum_{\sim x} F_1(x, \vec{X}_1)$, per $k = 1$, che equivale al messaggio inviato dal nodo fattore associato alla funzione F_1 ad x. Dato che anche F_1 è rappresentabile attraverso un grafo ad albero, si può scrivere senza perdere generalità:

$$F_1(x, \vec{X}_1) = f_1(x, x_1, .., x_L) \cdot G_1(x_1, \vec{X}_{11}) .. \cdot G_L(x_L, \vec{X}_{1L}), \tag{A.4}$$

assumendo che esistano L funzioni G_i e la funzione f_1, che è stata scritta diversamente per sottolineare la dipendenza diretta dalla variabile x.

Anche in questo caso i subset $\vec{X}_{1i}$ e $(x_1, .., x_L)$ sono una partizione per $\vec{X}_1$ e quindi, ancora una volta, applicando la proprietà distributiva:

$$\begin{aligned} \sum_{\sim x} F_1(x, \vec{X}_1) &= \sum_{\sim x} f_1(x, x_1, .., x_L) \cdot G_1(x_1, \vec{X}_{11}) .. \cdot G_L(x_L, \vec{X}_{1L}) = \\ &= \sum_{\{x_1, .., x_L\}} f_1(x, x_1, .., x_L) \prod_{i=1}^{L} \sum_{\sim x_i} G_i(x_i, \vec{X}_{1i}) \end{aligned} \tag{A.5}$$

Perciò, il messaggio inviato dal nodo f_1 ad x si ottiene dalla sommatoria $\sum_{\sim x}$ eseguita sul prodotto tra la funzione f_1 e il prodotto dei messaggi provenienti dai nodi $x_1, .., x_L$.

Ora si studi la natura di questi ultimi osservando il caso del messaggio $\sum_{\sim x_1} G_1(x_1, \vec{X}_{11})$ inviato dal nodo variabile x_1 al nodo fattore f_1. Analogamente ai casi precedenti la funzione G_1 è rappresentabile come un grafo ad albero e siccome in questo caso il riferimento è x_1, si operi similmente a come fatto per x. Si considerino allora un numero J di funzioni dipendenti da x_1, oltre a f_1, il cui prodotto forma G_1:

$$G_1(x_1, \vec{X}_{11}) = \prod_{j=1}^{J} H_j(x_1, \vec{Y}_j), \tag{A.6}$$

dove pure in questo caso il set di $\vec{Y}_j$ è una partizione di $\vec{X}_{11}$. Applicando gli stessi passaggi visti in precedenza si ottiene:

$$\sum_{\sim x_1} G_1(x_1, \vec{X}_1) = \prod_{j=1}^{J} \sum_{\sim x_1} H_j(x_1, \vec{Y}_j). \tag{A.7}$$

Da ciò si deduce che il messaggio inviato dal nodo x_1 al nodo f_1 è pari al semplice prodotto dei messaggi provenienti dai nodi fattori H_j.

In conclusione, è possibile affermare che un messaggio inviato da un nodo variabile corrisponde all'operazione di moltiplicazione dei messaggi provenienti da tutti i nodi fattori connessi, eccetto il destinatario. Invece, un messaggio inviato da un nodo fattore corrisponde ad una marginalizzazione del prodotto tra la funzione associata al nodo e il prodotto dei messaggi provenienti da tutti i nodi variabile connessi, eccetto il destinatario.

Questi risultati esplicitano la corrispondenza tra i nodi dei grafi fattoriali e le operazioni aritmetiche dei nodi degli expression trees.

Appendice B

BP sui grafi ad albero

In questa appendice è dimostrato il seguente teorema:

Teorema 1. *Si definisca il diametro di un grafo come il valore massimo tra le distanze minime fra ogni coppia di vertici. É possibile dimostrare per un grafo ad albero con diametro $\hat{d}$ che:*

1. *Indipendentemente dalla condizione iniziale, $\forall i, a$ per $t > \hat{d}$:*

$$\begin{aligned} \nu^t_{x_i \to f_a} &= \nu_{x_i \to f_a} \\ \hat{\nu}^t_{f_a \to x_i} &= \hat{\nu}_{f_a \to x_i}; \end{aligned} \tag{B.1}$$

2. *I punti fissi sono uguali esattamente ai marginali.*

Dimostrazione. Si riprenda la regola di aggiornamento vista nel capitolo 2:

$$\nu^t_{x_i \to f_a}(x_i) = \prod_{k \in \partial x_i \setminus a} \hat{\nu}^t_{f_k \to x_i}(x_i),$$

$$\hat{\nu}^{t+1}_{f_a \to x_i}(x_i) = \sum_{\sim x_i} [f_a(\vec{X}_a) \prod_{j \in n(f_a) \setminus i} \nu^t_{x_j \to f_a}(x_j)].$$

Si consideri un generico arco (i, a) e lo si studi direzionato $i \to a$. Si valuti quindi il sotto-grafo ad albero $G(i \to a)$ con apice in i, e si definisca il diametro del sotto-grafo come $\hat{d}(i \to a)$.

Si dimostrino i due punti del teorema per induzione, quindi si studi il caso unitario in cui $G(i \to a)$ è il solo nodo i. In questo caso la funzione associata al sotto-grafo è triviale e $\nu_{x_i \to f_a}(x_i)$=1 $\forall t \geq 1$, per cui i punti del teorema sono verificati.

Ora si considerino veri i punti del teorema per $\hat{d}(i \to a) \leq \tau$ e si prenda in considerazione il caso successivo $\hat{d}(i \to a) = \tau + 1$ per $t > \tau + 1$.

Il messaggio $\nu^{t+1}_{x_i \to f_a}$ è calcolabile grazie alle formule precedenti dai messaggi $\nu^{t}_{x_j \to f_k}(x_i)$ calcolati al tempo t. Siccome si ha $t > \tau + 1$, i sottografi associati a tali messaggi hanno al massimo diametro τ e rappresentano dei marginali esatti.

Sapendo ciò e sostituendo opportunamente è possibile verificare che anche $\nu^{t+1}_{x_i \to f_a}$ è il marginale esatto.

□

Appendice C

Messaggio da nodo variabile a nodo fattore

Lo scopo di questo appendice è esplicitare i passaggi algebrici che consentono di parametrizzare la prima equazione di BP standard in 3.21 nella prima equazione delle magnetizzazioni cave 3.28, nel caso in cui si considerino variabili di Ising.

Si consideri il messaggio inviato dal nodo variabile s_i al nodo fattore relativo ϵ_a:

$$\nu^t_{s_i\to\epsilon_a}(s_i) = \prod_{k\in\partial s_i\setminus a} \hat{\nu}_{\epsilon_k\to s_i}(s_i). \tag{C.1}$$

Riprendendo l'equazione 3.27, la magnetizzazione cava $m^t_{s_i\to\epsilon_a}$ si può scrivere anche come:

$$m^t_{s_i\to\epsilon_a} = \frac{\nu^t_{s_i\to\epsilon_a}(+1) - \nu^t_{s_i\to\epsilon_a}(-1)}{\nu^t_{s_i\to\epsilon_a}(+1) + \nu^t_{s_i\to\epsilon_a}(-1)}. \tag{C.2}$$

Inserendo l'espressione di $\nu^t_{s_i\to\epsilon_a}(s_i)$ in C.2 per i vari valori di s_i, si semplifichi l'equazione secondo i seguenti passaggi, dove saranno trascurati gli indici t per comodità.

Dimostrazione.

$$m_{s_i\to\epsilon_a} = \frac{1 - \frac{\nu_{s_i\to\epsilon_a}(-1)}{\nu_{s_i\to\epsilon_a}(+1)}}{1 + \frac{\nu_{s_i\to\epsilon_a}(-1)}{\nu_{s_i\to\epsilon_a}(+1)}} = \frac{1 - e^{-2x}}{1 + e^{-2x}} = tanh(x),$$

dove

$$x = \frac{1}{2} ln\left(\frac{\nu_{s_i\to\epsilon_a}(+1)}{\nu_{s_i\to\epsilon_a}(-1)}\right).$$

Utilizzando C.1:

$$x = \frac{1}{2} ln\left(\frac{\prod_{k\in\partial s_i\setminus a}\hat{\nu}_{\epsilon_k\to s_i}(+1)}{\prod_{k\in\partial s_i\setminus a}\hat{\nu}_{\epsilon_k\to s_i}(-1)}\right) = \frac{1}{2}\sum_{k\in\partial s_i\setminus a} ln\left(\frac{\hat{\nu}_{\epsilon_k\to s_i}(+1)}{\hat{\nu}_{\epsilon_k\to s_i}(-1)}\right)$$

e riscrivendo il termine nel logaritmo

$$x = \frac{1}{2} \sum_{k \in \partial s_i \setminus a} ln \left(\frac{1 + \frac{\hat{\nu}_{\epsilon_k \to s_i}(+1) - \hat{\nu}_{\epsilon_k \to s_i}(-1)}{\hat{\nu}_{\epsilon_k \to s_i}(+1) + \hat{\nu}_{\epsilon_k \to s_i}(-1)}}{1 - \frac{\hat{\nu}_{\epsilon_k \to s_i}(+1) - \hat{\nu}_{\epsilon_k \to s_i}(-1)}{\hat{\nu}_{\epsilon_k \to s_i}(+1) + \hat{\nu}_{\epsilon_k \to s_i}(-1)}} \right)$$

si ottiene

$$x = \sum_{k \in \partial s_i \setminus a} \frac{1}{2} ln \left(\frac{1 + \hat{m}_{\epsilon_k \to s_i}}{1 - \hat{m}_{\epsilon_k \to s_i}} \right) = \sum_{k \in \partial s_i \setminus a} artanh\, \hat{m}_{\epsilon_k \to s_i}$$

in cui

$$\hat{m}_{\epsilon_k \to s_i} = \frac{\hat{\nu}_{\epsilon_k \to s_i}(+1) - \hat{\nu}_{\epsilon_k \to s_i}(-1)}{\hat{\nu}_{\epsilon_k \to s_i}(+1) + \hat{\nu}_{\epsilon_k \to s_i}(-1)}.$$

Ricomponendo l'equazione iniziale si è ricavato:

$$m_{s_i \to \epsilon_a} = tanh \left[\sum_{k \in \partial s_i \setminus a} artanh(\hat{m}_{\epsilon_k \to s_i}) \right] \qquad \square$$

che conferma la prima equazione in 3.24.

Si noti in particolare che è stato possibile ritrovare tale equazione a prescindere dalla forma delle funzioni associate ai nodi fattori, poichè è stato solo sfruttato la natura di Ising delle variabili. Ciò significa che a prescindere dalla natura dell'Hamiltoniana che governa un sistema, è sufficiente che le variabili trattate siano binarie per riscrivere la prima equazione di BP in questa forma.

Appendice D

Messaggio da nodo fattore a nodo variabile

Lo scopo di questo appendice è esplicitare i passaggi algebrici che consentono di parametrizzare la seconda equazione di BP standard in 3.21 nella seconda equazione delle magnetizzazioni cave 3.28, nel caso in cui si considerino variabili di Ising.

I nodi variabili sono associati gli spin s_i, mentre i nodi fattori rappresentano, considerando l'Hamiltoniana 3.17, le funzioni non normalizzate $e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)} s_j}$, che sono indicate con il corrispondente fattore di coupling ϵ_a.

Il messaggio dal nodo fattore ϵ_a al nodo variabile è:

$$\hat{\nu}_{\epsilon_a \to s_i}(s_i) = \sum_{\sim s_i} [e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)} s_j} \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)], \tag{D.1}$$

dove per comodità è stato trascurato il tempo t.

Riprendendo l'equazione 3.27, la magnetizzazione cava $\hat{m}_{\epsilon_a\to s_i}$ si può scrivere come:

$$\hat{m}_{\epsilon_a\to s_i} = \frac{\hat{\nu}_{\epsilon_a\to s_i}(+1) - \hat{\nu}_{\epsilon_a\to s_i}(-1)}{\hat{\nu}_{\epsilon_a\to s_i}(+1) + \hat{\nu}_{\epsilon_a\to s_i}(-1)}. \tag{D.2}$$

Inserendo l'espressione di $\hat{\nu}_{\epsilon_a\to s_i}(s_i)$ in D.2 per i vari valori di s_i e seguendo i seguenti passaggi è possibile ottenere la seconda equazione in 3.28.

Dimostrazione.

$$\hat{m}_{\epsilon_a\to s_i} = \frac{\sum_{\{s_j\}}[e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)] - \sum_{\{s_j\}}[e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)]}{\sum_{\{s_j\}}[e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)] + \sum_{\{s_j\}}[e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)]}, \tag{D.3}$$

che si può anche riscrivere come:

$$\hat{m}_{\epsilon_a \to s_i} = \frac{\sum_{\{s_j\}}[(e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} - e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j}) \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)]}{\sum_{\{s_j\}}[(e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} + e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j}) \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)]}.$$

Ora si osservi il numeratore:

$$\sum_{\{s_j\}}[(e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j} - e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus i} s_j}) \prod_{j\in n(\epsilon_a)\setminus i} \nu_{s_j\to\epsilon_a}(s_j)].$$

Si sciolga nella sommatoria la variabile s_k per $s_k \in \{s_j\}$:

$$\begin{aligned}\sum_{\{s_j\}\setminus s_k} [\Big((e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} s_j} - e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} s_j})\nu_{s_k\to\epsilon_a}(1) + \\ +(e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} s_j} - e^{+\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} s_j})\nu_{s_k\to\epsilon_a}(-1)\Big) \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} \nu_{s_j\to\epsilon_a}(s_j)],\end{aligned} \tag{D.4}$$

il quale può essere riscritto come:

$$\sum_{\{s_j\}\setminus s_k}[(e^{\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} s_j} - e^{-\beta\epsilon_a \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} s_j})(\nu_{s_k\to\epsilon_a}(1) - \nu_{s_k\to\epsilon_a}(-1)) \prod_{j\in n(\epsilon_a)\setminus\{i,k\}} \nu_{s_j\to\epsilon_a}(s_j)]. \tag{D.5}$$

A questo punto il termine $(\nu_{s_k\to\epsilon_a}(1) - \nu_{s_k\to\epsilon_a}(-1))$ può essere portato fuori dalla sommatoria. Effettuando lo stesso procedimento per tutte le variabili s_j il numeratore diventa:

$$(e^{\beta\epsilon_a} - e^{-\beta\epsilon_a}) \prod_{j\in n(\epsilon_a)\setminus i} (\nu_{s_j\to\epsilon_a}(1) - \nu_{s_j\to\epsilon_a}(-1)).$$

Gli stessi passaggi si possono svolgere analogamente anche al denominatore, che quindi si può riscrivere come:

$$(e^{\beta\epsilon_a} + e^{-\beta\epsilon_a}) \prod_{j\in n(\epsilon_a)\setminus i} (\nu_{s_j\to\epsilon_a}(1) + \nu_{s_j\to\epsilon_a}(-1)).$$

Di conseguenza $\hat{m}_{\epsilon_a\to s_i}$ è uguale a:

$$\hat{m}_{\epsilon_a\to s_i} = \frac{(e^{\beta\epsilon_a} - e^{-\beta\epsilon_a}) \prod_{j\in n(\epsilon_a)\setminus i}(\nu_{s_j\to\epsilon_a}(1) - \nu_{s_j\to\epsilon_a}(-1))}{(e^{\beta\epsilon_a} + e^{-\beta\epsilon_a}) \prod_{j\in n(\epsilon_a)\setminus i}(\nu_{s_j\to\epsilon_a}(1) + \nu_{s_j\to\epsilon_a}(-1))},$$

a sua volta riscrivibile come:

$$\hat{m}_{\epsilon_a\to s_i} = \frac{(e^{\beta\epsilon_a} - e^{-\beta\epsilon_a})}{(e^{\beta\epsilon_a} + e^{-\beta\epsilon_a})} \prod_{j\in n(\epsilon_a)\setminus i} \frac{(\nu_{s_j\to\epsilon_a}(1) - \nu_{s_j\to\epsilon_a}(-1))}{(\nu_{s_j\to\epsilon_a}(1) + \nu_{s_j\to\epsilon_a}(-1))}.$$

Richiamando nuovamente l'equazione 3.27, si ha:

$$m_{s_j\to\epsilon_a} = \frac{\nu_{s_j\to\epsilon_a}(+1) - \nu_{s_j\to\epsilon_a}(-1)}{\nu_{s_j\to\epsilon_a}(+1) + \nu_{s_j\to\epsilon_a}(-1)},$$

e notando pure che il primo termine è una tangente iperbolica, si ritrova che il messaggio dal nodo fattore ϵ_a al nodo variabile s_i è:

$$\hat{m}_{\epsilon_a \to s_i} = tanh(\beta \epsilon_a) \prod_{j \in n(\epsilon_a) \setminus i} m_{s_j \to \epsilon_a},$$

che conferma la seconda equazione in 3.28(trascurando t). □

Si noti che a differenza del messaggio da nodo variabile a nodo fattore, la forma qui ritrovata è dipesa dalla natura dell'Hamiltoniana 3.17 che definisce le funzioni nelle formule di BP.

Appendice E

Funzione di partizione con repliche

Lo scopo di questa appendice è mostrare i passaggi che consentono di ricavare la funzione di partizione 5.23.

Si consideri la funzione di partizione:

$$Z(\beta, t, \gamma) = \sum_{\vec{\sigma}^\star} e^{y\, log \sum_{\vec{\sigma}'} e^{-\beta E(\vec{\sigma}') - \gamma d(\vec{\sigma}^\star, \vec{\sigma}')}}, \tag{E.1}$$

dove $\vec{\sigma}^\star$ indica una configurazione di riferimento e $\sum_{\vec{\sigma}^\star}$ la somma sull'insieme delle configurazioni di riferimento.

Per semplicità si consideri solo l'argomento della sommatoria; per il quale vale l'uguaglianza:

$$e^{y\, log \sum_{\vec{\sigma}'} e^{-\beta E(\vec{\sigma}') - \gamma d(\vec{\sigma}^\star, \vec{\sigma}')}} = \Big(\sum_{\vec{\sigma}'} e^{-\beta E(\vec{\sigma}') - \gamma d(\vec{\sigma}^\star, \vec{\sigma}')} \Big)^y.$$

Assumendo y intero non negativo, si può scrivere:

$$\Big(\sum_{\vec{\sigma}'} e^{-\beta E(\vec{\sigma}') - \gamma d(\vec{\sigma}^\star, \vec{\sigma}')} \Big)^y = \prod_{a=1}^{y} \sum_{\vec{\sigma}^a} e^{-\beta E(\vec{\sigma}^a) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^a)}, \tag{E.2}$$

che equivale a considerare y repliche identiche del sistema $\vec{\sigma}'$. L'identità nasce assumendo che le sommatorie $\sum_{\vec{\sigma}^a}$ siano uguali per ogni valore di a.

Sciogliendo la produttoria si ha:

$$\Big(\sum_{\vec{\sigma}^1} e^{-\beta E(\vec{\sigma}^1) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^1)} \Big) \Big(\sum_{\vec{\sigma}^2} e^{-\beta E(\vec{\sigma}^2) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^2)} \Big) .. \Big(\sum_{\vec{\sigma}^y} e^{-\beta E(\vec{\sigma}^y) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^y)} \Big), \tag{E.3}$$

pari a:

$$\sum_{\vec{\sigma}^1} \sum_{\vec{\sigma}^2} .. \sum_{\vec{\sigma}^y} \prod_{a=1}^{y} e^{-\beta E(\vec{\sigma}^a) - \gamma d(\vec{\sigma}^\star, \vec{\sigma}^a)}.$$

Definendo $\sum_{\vec{\sigma}^1} \sum_{\vec{\sigma}^2} .. \sum_{\vec{\sigma}^y} = \sum_{\{\vec{\sigma}^a\}}$ e svolgendo la produttoria per gli esponenziali si ricava:

$$\sum_{\vec{\sigma}^a} e^{-\beta \sum_{a=1}^{y} E(\{\vec{\sigma}^a\}) - \gamma \sum_{a=1}^{y} d(\vec{\sigma}^\star, \vec{\sigma}^a)}, \tag{E.4}$$

che inserita nella formula della funzione di partizione conduce all'equazione 5.23:

$$Z(\beta, t, \gamma) = \sum_{\vec{\sigma}^\star} \sum_{\{\vec{\sigma}^a\}} e^{-\beta \sum_{a=1}^{y} E(\vec{\sigma}^a) - \gamma \sum_{a=1}^{y} d(\vec{\sigma}^\star, \vec{\sigma}^a)}.$$

Appendice F

Contributo extra delle repliche

Questo appendice ha lo scopo di dimostrare la formula 5.34 che rappresenta il messaggio extra inviato ai nodi variabili dei pesi σ_i dall'insieme delle repliche.

Si riconsideri la figura 5.9, che per comodità si riporta di seguito.

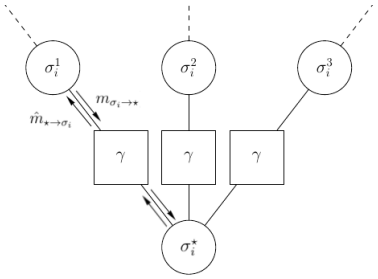

Figura F.1: Connessione di tre repliche del peso σ_i.

Come visto nella sezione 5.3, l'interazione tra ogni replica identica $\vec{\sigma}^a$ e la replica di riferimento $\vec{\sigma}^\star$ è espressa nella distribuzione di probabilità da $e^{-\gamma \vec{\sigma}^\star \cdot \vec{\sigma}^a}$. Questo termine è associato ad un contributo nell'energia proporzionale a $\gamma \sum_i \sigma_i^\star \sigma_i^a$, che è nella stessa forma dei contributi nell'Hamiltoniana dei problemi di Spin Glasses 3.17.

Come spiegato in 3.3.2, ciò implica che i messaggi scambiati tra i nodi sono sintetizzabili nelle equazioni 3.28, ovvero (evitando di riportare anche t):

$$\begin{aligned} m_{s_i \to \epsilon_a} &= tanh\left[\sum_{k \in \partial s_i \setminus a} artanh(\hat{m}_{\epsilon_k \to s_i}) \right] \\ \hat{m}_{\epsilon_a \to s_i} &= tanh(\beta \epsilon_a) \prod_{j \in n(\epsilon_a) \setminus i} m_{s_j \to \epsilon_a}, \end{aligned} \tag{F.1}$$

dove in questo caso il nodo fattore è γ e le variabili s_i sono σ_i^a e $\sigma_i^\star$.

Si consideri ora $\sigma^{a'}$ una tra le σ^a, uno dei pesi i e si cerchi il contributo extra dato a $\sigma_i^{a'}$ dalle rimanenti $\{\sigma_i^a\}_{a \neq a'}$.

Il messaggio $m_{a\to\star_a}$ inviato da un σ_i^a al relativo nodo fattore γ include tutte le informazioni provenienti dal restante grafo fattoriale. Questo contributo è l'unico entrante in γ, quindi il messaggio $\hat{m}_{\star_a\to *}$ da γ al peso di riferimento $\sigma_i^\star$ è, sfruttando la seconda equazione in F.1:

$$\hat{m}_{\star_a\to *} = tanh(\gamma)\, m_{a\to\star_a}.$$

A questo punto il messaggio che $\sigma_i^\star$ passa al nodo fattore γ (connesso a $\sigma^{a'}$) è, adattando la prima equazione in F.1, uguale a:

$$m_{*\to\star_{a'}} = tanh\left[\sum_{a\in\partial *\setminus a'} artanh(\hat{m}_{\star_a\to *})\right].$$

Il contributo extra ricevuto da $\sigma^{a'}$ dalle $\{\sigma_i^a\}_{a\neq a'}$ è infine ottenuto considerando il messaggio $m_{\star_{a'}\to a'}$ inviato dal corrispondente nodo fattore γ a $\sigma^{a'}$, pari a (usando ancora la seconda equazione in F.1):

$$\hat{m}_{\star_{a'}\to a'} = tanh(\gamma)\, m_{*\to\star_{a'}}.$$

Semplificando tutti i messaggi intermedi $\hat{m}_{\star_{a'}\to a'}$ diventa:

$$\hat{m}_{\star_{a'}\to a'} = tanh(\gamma)\left(tanh\left[\sum_{a\in\partial *\setminus a'} artanh(tanh(\gamma)\, m_{a\to\star_a})\right]\right).$$

In conclusione, siccome le repliche $\{\sigma_i^a\}$ sono considerate identiche e i messaggi scambiati in essi uguali il contributo extra diviene semplicemente:

$$\hat{m}_{\star\to\sigma_i} = \Big[tanh(y-1)artanh(tanh(\gamma)\, m_{\sigma_i\to\star})\Big]tanh(\gamma), \tag{F.2}$$

che conferma l'equazione 5.34.

Bibliografia

[1] R. Balescu. *Equilibrium and nonequilibrium statistical mechanics*. 1975.

[2] K. Huang. Statistical Mechanics, 2nd Edition, 1987.

[3] R. K. Pathria. Statistical Mechanics. *Statistical Mechanics*, 1996.

[4] C. M. Bishop. Neural networks for pattern recognition. *Journal of the American Statistical Association*, 1995.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning*. 2007.

[6] S. Haykin. *Neural Networks and Learning Machines*. 2008.

[7] F. R. Kschischang and et al. Factor Graphs and Sum Product Algorithm. *IEEE Transactions on Information Theory*, 2001.

[8] H. E. Kyburg and J. Pearl. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. *The Journal of Philosophy*, 1991.

[9] J. S. Yedidia and et al. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 2005.

[10] M. Mézard and A. Montanari. *Information, Physics, and Computation*. 2009.

[11] D. J. C. Mackay. Information Theory , Inference , and Learning Algorithms. *Learning*, 2003.

[12] D. J. Thouless and et al. Solution of 'Solvable model of a spin glass'. *Philosophical Magazine*, 1977.

[13] H. Nishimori. *Statistical Physics of Spin Glasses and Information Processing An Introduction*. 2001.

[14] M. Mézard and G. Parisi. The Cavity Method at Zero Temperature. *Journal of Statistical Physics*, 2003.

[15] G. Parisi. Mean field theory of spin glasses: statics and dynamics. 2007.

[16] Y. Kabashima. Propagating beliefs in spin-glass models. *Journal of the Physical Society of Japan*, 2003.

[17] Y. Kabashima. A CDMA multiuser detection algorithm on the basis of belief propagation. *Journal of Physics A: Mathematical and General*, 2003.

[18] Y. Kabashima. Replicated Bethe free energy: A variational principle behind survey propagation. *Journal of the Physical Society of Japan*, 2005.

[19] J. P. Neirotti and D. Saad. Improved message passing for inference in densely connected systems. *Europhysics Letters*, 2005.

[20] T. L. H. Watkin and et al. The statistical mechanics of learning a rule. *Rev. Mod. Phys.*, 1993.

[21] A. L. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 1992.

[22] W. Krauth and M. Mezard. Storage capacity of memory networks with binary coupling. *J. Phys (France)*, 1989.

[23] A. Engel and C. Van den Broeck. *Statistical Mechanics of Learning*. 2012.

[24] R. Monasson and R. Zecchina. Weight space structure and internal representations: A direct approach to learning and generalization in multilayer neural networks. *Physical Review Letters*, 1995.

[25] C. Baldassi and et al. Efficient supervised learning in networks with binary synapses. *Proceedings of the National Academy of Sciences*, 2007.

[26] H. Huang and H. Zhou. Combined local search strategy for learning in networks of binary synapses. 2011.

[27] C. Baldassi. Generalization learning in a perceptron with binary synapses. 2012.

[28] C. Baldassi and A. Braunstein. A Max-Sum algorithm for training discrete neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2015.

[29] C. Baldassi and et al. Subdominant Dense Clusters Allow for Simple Learning and High Computational Performance in Neural Networks with Discrete Synapses. *Physical Review Letters*, 2015.

[30] S. Franz and G. Parisi. Recipes for Metastable States in Spin Glasses. *Journal de Physique I, EDP Sciences*, 1995.

[31] H. Huang and Y. Kabashima. Origin of the computational hardness for learning with binary synapses. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 2014.

[32] C. Baldassi and et al. Local entropy as a measure for sampling solutions in constraint satisfaction problems. *Journal of Statistical Mechanics: Theory and Experiment*, 2016.

[33] C. Baldassi and et al. Learning may need only a few bits of synaptic precision. *Physical Review E*, 2016.

[34] A. S. Sara and W. Ole. Optimal perceptron learning: an online Bayesian approach. 1998.

[35] Y. S. Xiong and et al. Storage capacity of a fully-connected parity machine with continuous weights. *Journal of Physics A: Mathematical and General*, 1998.

[36] R. Monasson and R. Zecchina. Learning and Generalization Theories of Large Committee Machines. *Modern Physics Letters B*, 1995.

[37] C. Baldassi and et al. Unreasonable Effectiveness of Learning Neural Networks: From Accessible States and Robust Ensembles to Basic Algorithmic Schemes. 2016.

[38] A. Braunstein and R. Zecchina. Learning by message passing in networks of discrete synapses. *Physical Review Letters*, 2006.

[39] B. Alberts and et al. *Molecular Biology of the Cell*. 2017.

[40] J. Shendure and H. Ji. Next-generation DNA sequencing. *Nature Biotechnology*, 2008.

[41] S. Behjati and P.S. Tarpey. What is next generation sequencing? *Archives of disease in childhood - Education & practice edition*, 2013.

[42] Z. Šidák. Rectangular Confidence Regions for the Means of Multivariate Normal Distributions. *Journal of the American Statistical Association*, 1967.

[43] B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *BBA - Protein Structure*, 1975.

[44] J. Snoek and et al. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'12, 2012.

[45] B. A. Moore and G. J. McLachlan. Discriminant Analysis and Statistical Pattern Recognition. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 2006.

[46] M. Kuhn and K. Johnson. *Applied predictive modeling*. 2013.

[47] L. Breiman and et al. *Classification and regression trees*. 2017.

[48] C. Baldassi and et al. Efficiency of quantum vs. classical annealing in nonconvex learning problems. 2017.

[49] H. C. Nguyen and et al. Inverse statistical problems: from the inverse Ising problem to data science. *Advances in Physics*, 2017.

[50] C. Baldassi and et al. From inverse problems to learning: A Statistical Mechanics approach. 2018.

[51] C. Baldassi and et al. Role of Synaptic Stochasticity in Training Low-Precision Neural Networks. *Physical Review Letters*, 2018.

[52] L. Saglietti and et al. From statistical inference to a differential learning rule for stochastic neural networks. *Interface Focus*, 2018.