

Thesis for the Degree of Master of Science

Quantum Information Theory for Machine Learning

Rikard Wadman



Department of Mathematical Sciences
Division of Algebra and Geometry
CHALMERS UNIVERSITY OF TECHNOLOGY
SE – 412 96 Gothenburg, Sweden
Gothenburg, October 2019

Quantum Information Theory for Machine Learning
Rikard Wadman
rikard.wadman@gmail.com

© Rikard Wadman, 2019.

Supervisor & Examiner: Daniel Persson, Department of Mathematical Sciences

Master's Thesis 2019
Department of Mathematical Sciences
Division of Algebra and Geometry
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Stylized image of a tree tensor network contracted with a pure tensor, known from quantum mechanics as a product state and from the machine learning context as a local feature map.

Typeset in L^AT_EX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2019

Quantum Information Theory for Machine Learning
A study of machine learning in vast product spaces using tensor networks
RIKARD WADMAN
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

The remarkable successes of machine learning and of deep learning in particular during the last decade have caused an explosive growth of interest in the field. Meanwhile, there are still significant gaps in our understanding of the processes involved, making the area a very promising topic for theoretical investigation. A particularly interesting idea that has received a lot of attention recently is the claim that the successive transformations performed by deep neural networks behave similarly to the renormalization group flows of statistical mechanics. In the light of this it is natural also to consider numerical renormalization algorithms as interesting candidates for performing general machine learning. It turns out that both the DMRG and the more recent Entanglement Renormalization algorithm from numerical quantum mechanics are quite well suited for this purpose. Both of these algorithms are most naturally described using the language of tensor networks, which are graph based representations of multilinear tensors, typically used for the description of quantum states. This thesis discusses machine learning with tensor networks from a holistic perspective and makes a review of some of the recent work on the subject.

Also of significant interest is the study of expressive power of neural networks. A recent proposal suggests employing quantum entanglement entropy as a measure of a models ability to represent complex correlations between input regions. We study the interpretability and implications of such a measure as well as its relations to the quantum version of the max-flow/min-cut theorem, which relates the entanglement entropy of a tensor network state to the minimal cut in its graph. A generalization of said theorem is found, leading us to alternate, and very simple, proofs of some already known scaling laws of quantum entanglement in Boltzmann machines and convolutional arithmetic circuits, which are derivative of standard convolutional neural networks.

Keywords: tensor networks, machine learning, quantum information, multilinear algebra, network theory, Convolutional arithmetic circuits, Boltzmann machines, entanglement, DMRG, MERA

Acknowledgements

As with most other things in life, this thesis could not have come together without the help and encouragement of a significant number of people. First and foremost, I would like to give my sincere gratitude to my supervisor Daniel Persson for great help and many interesting discussions on everything from fundamental mathematics and statistical mechanics to machine learning and life in general. I find your love of research, breadth of interests and welcoming attitude highly encouraging.

Jimmy Aronsson and Olof Carlsson also deserves great thanks, Jimmy for being so inviting at the start of my work and for helpful discussions, and Olof for providing such an excellent opposition on this thesis, which in the end became quite lengthy. I hope it was not too painful!

A special thanks goes to my fellow students Björn Eurenus, Michael Högberg and Daniel Erkensten for having the patience to let me ramble at length trying to explain parts of my thesis work. It has truly been a great help for my understanding. I am very sorry for taking so much of your time, but at the same time very grateful for you letting me do so.

Thank you also to the rest of the students taking the quantum theoretical track at Physics and Astronomy, for the almost too nice lunch breaks and for pulling me to campus, and to the musicians in the bands KiRA and Olimpias Orkan for getting me away from there from time to time. I am very grateful also to all my friends and family, who make life enjoyable, and who have been so encouraging on this journey of mine. Not least to Maria, my partner in life for the past six years for all your love, and for enduring all the late nights and talk of theoretical physics and mathematics. I hope you will allow me to continue boring you for many more years to come.

Finally, I would like to thank you, the reader, for deciding to pick up this work. I sincerely hope you will find it an enjoyable read, and that you have oversight with its inevitably numerous shortcomings.

Rikard Wadman, Göteborg, September 2019

Contents

List of Figures	xiii
Notation	xv
Abbreviations	xvii
1 Introduction	1
I Quantum theory & Mathematical foundations	3
2 Classical information theory	5
2.1 Shannon entropy	5
2.1.1 Joint entropies	7
2.1.2 Relative entropy/Kullback-Liebler divergence	8
2.1.3 Fisher information matrix	8
2.1.4 Mutual information	9
2.2 Rényi entropy	9
2.2.1 Rényi divergence	10
2.2.2 Rényi mutual information	10
3 Quantum information theory	11
3.1 Quantum theory – the bare minimum	11
3.2 Composite quantum mechanical systems	12
3.3 Schmidt decomposition	13
3.4 The density matrix	13
3.5 Measures of entanglement	15
3.5.1 Schmidt number	15
3.5.2 von Neumann entropy	15
3.5.3 Quantum Rényi entropy	16
3.5.4 Equal entropy of subsystems	17
3.5.5 Quantum mutual information	17
4 Graph theory	19
4.1 Types of graphs	19
4.2 Paths, cuts and cliques	20
4.3 Max flow/min cut and Menger’s theorem	21
4.4 Vertex limited flows	22
5 Tensor networks	25

5.1	An introduction	25
5.1.1	Merging edges	26
5.1.2	Taking derivatives	27
5.2	Entanglement in tensor networks	28
5.2.1	Quantum max flow/min cut	28
5.2.2	Quantum max flow/min cut with restricted tensors	31
5.2.3	Quantum max flow and entanglement entropy	34
5.3	Matrix product states	35
5.3.1	Gauge symmetry	35
5.3.2	Entanglement	36
6	Numerical renormalization group methods in quantum mechanics	37
6.1	Numerical real-space renormalization group	37
6.2	Density matrix renormalization group	38
6.2.1	Infinite lattice DMRG	39
6.2.2	Finite lattice DMRG	39
6.2.3	DMRG and MPS	40
6.3	Entanglement renormalization	41
II	Machine learning	43
7	Machine learning	45
7.1	Statistical learning theory	45
7.1.1	Supervised learning	45
7.1.2	Unsupervised learning	46
7.2	Artificial neural networks	47
7.2.1	Feedforward neural networks	47
7.3	Convolutional neural networks	48
7.3.1	Convolutions	49
7.3.2	The structure of a CNN	49
7.4	Structured probabilistic models	50
7.4.1	Bayesian networks	50
7.4.2	Markov random field	51
7.5	Boltzmann machines	52
7.5.1	Restricted Boltzmann machines	53
7.5.2	Deep belief networks	53
7.5.3	Deep Boltzmann machines	54
7.6	Kernel learning	54
7.6.1	Support vector machines	55
III	Quantum theory for machine learning	59
8	Tensor network learning	61
8.1	Tensor product Hilbert spaces in machine learning	61
8.1.1	Square-integrable functions	61
8.1.2	Local feature maps	62
8.1.3	Connecting to tensor networks	62
8.2	Variational tensor network learning	63

8.2.1	Gradient descent	64
8.2.2	Generalized DMRG	64
8.2.3	Supervised learning with MPS	64
8.3	Convolutional arithmetic circuits	65
8.3.1	Tensor network formulation	66
8.3.2	nCACs	67
8.4	Unsupervised coarse graining	68
8.4.1	The algorithm	68
8.4.2	Projection error	70
9	Quantum entanglement in machine learning	73
9.1	Rényi entropy of TNL models	74
9.2	Boltzmann machines	75
9.2.1	Mapping to tensor networks	75
9.2.2	Boltzmann machines as MPS	76
9.2.3	Area law of entanglement for local RBMs	77
9.2.4	Area law of entanglement for local DBMs	77
9.2.5	Volume law of entanglement for long range BMs and RBMs	78
9.3	Convolutional arithmetic circuits	78
9.3.1	Depth efficiency	78
10	Discussion	81
10.1	Numerical quantum mechanics for machine learning	81
10.2	Entanglement analysis of ML architectures	82
10.3	Quantum max-flow/min-cut	83
	Bibliography	85

List of Figures

4.1	Visual representations of closed and open graphs respectively. Each point corresponds to a vertex and each line to an edge.	20
6.1	Illustration of the blocking procedure of the lattice in (a) infinite and (b) finite DMRG. Each line is to be interpreted as a step in the algorithm, which runs from top to bottom.	40
7.1	Graphical representations of feedforward neural networks. The graph of a single neuron with three inputs is shown in (a), an example of an arbitrary FNN is shown in (b) and (c) shows a layered FNN.	48
7.2	Shown in (a) is a Bayesian Network representing the probability distribution $p(x,y,z,w) = p(x)p(y x)p(z x,y)p(w z)$. In (b) is a Markov random field representing the probability distribution $p(x,y,z,w) = \frac{1}{Z}e^{-E_1(x,y,z)-E_2(z,w)}$	51
7.3	Panels (a), (b) and (c) shows graphical representations of a small dense restricted Boltzmann machine, deep belief network and deep Boltzmann machine respectively. The RBM and DBM are represented as Markov random fields, while the DBN is represented as a Markov random field in the two uppermost layers but a Bayesian network in all lower layers.	53
7.4	A map $\Phi(x)$ taking the input patterns $x_i \in \mathcal{X}$ (left panel) with $y_i = 1$ (filled circles) or -1 (open circles) to an inner product space \mathcal{H} (right panel) where the classes are linearly separable. ¹	55
8.1	The tensor network representation of $h(x)$ for a 1D convolutional arithmetic circuit with four layers and $S = K = 3$. Here W^1, W^2, W^3 have the internal structure shown in (8.30), while W^4 may be any matrix of appropriate dimensionality.	67
9.1	The Boltzmann machine represented by the Markov random graph in (a) models the same function as the tensor network in (b). The light grey dots represent hidden units, the dark blue dots are visible units and the green diamonds are the matrices $M^{(ij)}$. Hidden and visible units h_i and v_j in the random graph are replaced by tensors $\Lambda_h^{(i)}$ and $\Lambda_v^{(j)}$ respectively in the tensor network.	76

Notation

Sets

$ A $	The cardinality of the set A
$[N]$	The set $\{1, 2, \dots, N\}$
$[A]^k$	The set of unordered k -tuples in A , i.e. the set $\{\{a_1, \dots, a_k\}; a_i \in A\}$

Morphisms

$\text{hom}(V, W)$	A homomorphism, from V to W . In the thesis, V and W will almost always be Hilbert spaces, in which case $\text{hom}(V, W)$ is a linear map from V to W .
$\text{end}(V)$	An endomorphism on V . Equivalent to $\text{hom}(V, V)$.
$\text{id}(V)$	The identity map on V .

Dirac notation

$ \psi\rangle$	A vector in some Hilbert space \mathcal{H} . Typically used for describing a quantum state.
$\langle\psi $	The dual of $ \psi\rangle$.
$\langle\varphi O \psi\rangle$	An inner product between $\langle\varphi $ and $O \psi\rangle$, with O being a linear operator on \mathcal{H} . Equivalent to $\langle O\psi, \varphi\rangle$

Graphs

$G = (V, E)$	A graph with vertices V and edges E .
dv	The set of edges connected to the vertex $v \in V$.
∂e	The set of vertices connected to the edge $e \in E$.
\bar{E}	The set of internal edges (edges for which both ends connects to vertices) in an open graph.
∂E	The set of dangling edges (edges for which only one end connects to a vertex) in an open graph. $E = \bar{E} \cup \partial E$.

Tensors

T_{i_1, \dots, i_n}	An n -mode tensor. In general T takes values in an n -fold tensor product Hilbert space $\bigotimes_{j=1}^n \mathcal{H}_j$.
$\text{Diag}(\mathbf{v})$	A diagonal tensor, equal to \mathbf{v}_i if all indices are equal to i , otherwise zero.
δ_{i_1, \dots, i_n}	The n -mode Kronecker delta. Equal to the n -mode $\text{Diag}(\{1, 1, 1, \dots\})$.

Abbreviations

ANN	Artificial neural network
BM	Boltzmann machine
CAC	Convolutional arithmetic circuit
CNN	Convolutional neural network
DBM	Deep Boltzmann machine
DBM	Deep belief network
DL	Deep learning
DMRG	Density matrix renormalization group
DNN	Deep neural network
FNN	Feedforward neural network
MERA	Multiscale entanglement renormalization ansatz
ML	Machine learning
MPS	Matrix product state
nCAC	Near/Non-convolutional arithmetic circuit
QM	Quantum mechanics
PCA	Principal component analysis
RBM	Restricted Boltzmann machine
RG	Renormalization group
SVD	Singular value decomposition
SVM	Support vector machine
TN	Tensor network
TNL	Tensor network learning
TTN	Tree tensor network

1

Introduction

Machine learning (ML) is a field of research that has seen an enormous rise of interest over the last couple of years. Despite this, the field itself is hardly new; researchers have been interested in creating computer programmes which learn from experience ever since the birth of the computer in the mid 20th century, with notable examples being the invention of artificial neural networks [1] and experiments on Hebbian learning [2, 3].

The successes were rather limited early on, since the amount of available data as well as the computational resources at the time were far too small to allow for training of sufficiently complex models. However, with the continuous exponential growth in computational power, the introduction of the internet and the corresponding explosion of data, training of increasingly multi-layered, or *deep*, neural networks have become possible, spawning Deep learning (DL) as a new sub-field of ML and leading to unprecedented performance.

Particularly in the current decade, progress has been truly phenomenal – DL algorithms are now nearing human performance at tasks such as image recognition [4], speech recognition [5] and machine translation [6], and is being applied successfully to an increasing range of other problems (see e.g. [7] for a quick overview of the state of the art in ML).

While many practitioners have a good theoretical understanding, there is also a large degree of experimentation involved, and competing models are (and should be) judged based on their empirical merits. This has led to an interaction between the theoretical and experimental sides of ML much akin to that of particle physics in the mid 20th century, with significant flux of ideas between the two. On the theoretical side, the understanding of the processes involved is gradually unfolding, but the picture is still far from complete.

Machine learning has a long history of interdisciplinarity. While it is often seen as a subject within computer science, it is tightly tied to the mathematical branches of statistics, information theory and optimization, and many of the core ideas in the field, such as neural networks and convolutional networks are heavily inspired by neurobiological research. It comes as no surprise then that the ML community has also received significant inspiration from various branches of physics, both in terms of tools for model building and theoretical ideas which are applied to improve our understanding of the subject. One example of the former is the Boltzmann machines used for unsupervised learning, which model multivariate probability distributions identically to the Ising model of statistical mechanics. Variations of these were amongst the first models to permit training of deep architectures[8].

A significant example of ideas from physics being employed as analytical tools is the still quite controversial[9–12] interpretation of the training of deep neural networks as equivalent to finding a variational renormalization group (RG) transform, which was originally suggested in 2014 by Mehta and Schwab[13]. In their paper they specialize to deep belief networks, the hidden layers of which they argue behave equivalently to the coarse-graining of an Ising model.

The next thing to do, once this connection has been made, is of course to ask oneself

if this potential correspondence can be exploited, and in particular whether there are RG procedures which can be directly employed for ML. This does indeed turn out to be the case; especially the DMRG[14, 15] and entanglement renormalization (more known by the related multiscale entanglement renormalization ansatz, or MERA)[16] procedures of numerical condensed matter physics have been successfully employed for ML in different contexts[17–23]. Understanding these two RG procedures and their applications is one of the main goals of this thesis.

We are also taking a more general look at the applicability of quantum theory to machine learning. A particularly interesting topic is the use of quantum entropy as a quantifier of the expressivity of a neural network, suggested in this context by Levine et al[24].

As a finishing note, we remark that, although the focus in this thesis is on the applicability of quantum theory to machine learning, there are significant efforts being made both to apply ML to simulations of quantum systems, and to implement ML algorithms on quantum computers. These subjects share a common theoretical framework, and are heavily interrelated to the extent that single papers sometimes deal simultaneously with several of them. It is hence likely that developments in one area are also relevant to the other. In particular, a motivation for studying the quantum-inspired ML algorithms discussed above is that some of them may prove to be easier to implement on a quantum computer than standard ML models.

The goal of this thesis is to understand, from a theoretical viewpoint, the applications of quantum information theory and numerical renormalization group methods of quantum many-body systems to machine learning. The first six chapters are devoted to a bottom-up development of the underlying quantum theory and related mathematical constructions, starting with information and quantum information theory (chapters 2 and 3), continuing with a primer on graph theory (ch. 4) and an introduction to tensor networks (ch. 5), which are essential to understand the numerical renormalization group methods (ch. 6). The second part of the thesis is then a (relatively) quick walkthrough of relevant machine learning theory (ch. 7). Most of what is covered in the introductory chapters is standard material for the respective subjects, and the reader may hence wish to skip chapters with which they are already familiar. We do, however, in chapter 4 make a non-standard (but probably not original) extension of Menger’s theorem, which we then utilize to derive a slight generalization of the quantum max-flow/min-cut theorem of [25] in chapter 5. This generalization is to the best of our knowledge not previously proposed.

The third and main part of the thesis introduces the applications of quantum theory to ML discussed above and is divided into two chapters.

Chapter 8 deals with ML models which are either directly constructed from or can be interpreted as functions on the same form as quantum many-body wave functions. A coherent theoretical framework is developed in terms of L^2 functions and given an alternate interpretation in the context of kernel learning. A selection of ML models are then presented.

Chapter 9 deals with the usage of quantum entanglement entropy as a measure of the expressivity of a network. It gives a detailed view of how and when it is relevant and how it is to be interpreted. In addition, its original application to so-called convolutional arithmetic circuits, as well as studies of the scaling of quantum entropy in Boltzmann machines are explored.

The thesis concludes with a discussion of the developments in chapters 5, 8 and 9 and of possible directions for future research.

Part I

Quantum theory & Mathematical foundations

2

Classical information theory

Information theory is a mathematical theory invented in 1948 by Claude Shannon [26] as a tool to optimise the encoding of information in messages. The tools of information theory has subsequently found extensive use in both quantum physics and deep learning, and is hence a good place to start our theoretical developments. Much of this material can be found in any of [27–30], all of which are excellent materials to consult for a deeper understanding of the subject.

2.1 Shannon entropy

Consider a random variable X taking values in the finite set \mathcal{X} and following a distribution $p_X(x)$. To quantify the surprise, or information of observing $X = x$, we may introduce a measure as

$$i(x) \equiv -\log p_X(x). \quad (2.1)$$

This has several properties that couples nicely to the intuitive concepts of information. It is nonnegative, continuous and monotonic in p_X , $p(x) \rightarrow 0$ implies $i(x) \rightarrow \infty$ and conversely $p(x) = 1$ implies $i(x) = 0$. It is also additive; for two independently distributed random variables X and Y ,

$$i(x, y) \equiv -\log p_{X,Y}(x, y) = -\log p_X(x) - \log p_Y(y) = i_X(x) + i_Y(y). \quad (2.2)$$

While the existence of a well defined and intuitive measure of the information of individual events is interesting in its own right, it is often much more useful to consider the expected information, or uncertainty, in the random variable itself. This was first done with this particular information measure by Shannon [26] who defined the *entropy* of a random variable as the expected value of its information,

$$S(X) \equiv \mathbb{E}[i(X)] = -\sum_x p(x) \log p(x), \quad (2.3)$$

defining $0 \log 0 = 0$ for continuity of $x \log x$ at $x \geq 0$.

This entity has been shown to be unique (up to normalization) by several authors from slightly different sets of postulates [26, 31]. One particularly appealing postulation which we will present with slight modification is due to Rényi [32] and uses the language of generalized probability distributions. These he defines as probability distributions $p(x)$ which don't necessarily sum to one;

$$\sum_x p(x) \leq 1. \quad (2.4)$$

Now, let Δ be the family of all generalized finite probability distributions, i.e. the family of all sets on the form $\{p_1, \dots, p_n\}$ with $p_i \geq 0$ and $\sum_i p_i \leq 1$. Expressing the Shannon entropy as a function

$$S : \Delta \rightarrow \mathbb{R}, \quad (2.5)$$

Rényi's postulization for entropy in base $k \in (1, \infty)$ is:

1. $S(\{p\})$ is a continuous function of p .
2. $S(\{1/k\}) = 1$ (Originally, Rényi demanded $k = 2$, but we want to allow any $k \in (1, \infty)$ since this is just a normalization).
3. Let $\mathcal{P}, \mathcal{Q} \in \Delta$, and define $\mathcal{P} \times \mathcal{Q} \equiv \{pq; p \in \mathcal{P}, q \in \mathcal{Q}\}$. Then $S(\mathcal{P} \times \mathcal{Q}) = S(\mathcal{P}) + S(\mathcal{Q})$.
4. If $\mathcal{P}, \mathcal{Q} \in \Delta$ and $W(\mathcal{P}) + W(\mathcal{Q}) \leq 1$, with $W(\mathcal{P}) \equiv \sum_{p \in \mathcal{P}} p$ being the weight of \mathcal{P} , then

$$S(\mathcal{P} \cup \mathcal{Q}) = \frac{W(\mathcal{P})S(\mathcal{P}) + W(\mathcal{Q})S(\mathcal{Q})}{W(\mathcal{P}) + W(\mathcal{Q})}. \quad (2.6)$$

To prove that this leads to an uniquely determined entropy, we need the following lemma

Lemma 2.1. Let f be a (completely) additive arithmetic function, that is, a function $f : \mathbb{N} \rightarrow \mathbb{C}$ following $f(mn) = f(m) + f(n) \forall m, n \in \mathbb{N}$ and for which $f(n+1) - f(n) \xrightarrow{n \rightarrow \infty} 0$. Then

$$f(n) = K \log n,$$

for some $K \in \mathbb{C}$.

Proof. See e.g. [32, 33]. □

Theorem 2.1. (Uniqueness of Shannon entropy) [32]

Let $S : \Delta \rightarrow \mathbb{R}$ follow the above postulates. Then

$$S(\mathcal{P}) = -\frac{1}{W(\mathcal{P})} \sum_i p_i \log_k p_i,$$

where $k \in (1, \infty)$. For ordinary distributions, this reduces to $S = -\sum_i p_i \log_k p_i$.

Proof. Let $s(p) \equiv S(\{p\})$, for $p \leq 1$. By postulate 3, we have $s(pq) = s(p) + s(q)$. For $n, m \in \mathbb{N}$, this gives

$$s\left(\frac{1}{mn}\right) = s\left(\frac{1}{m}\right) + s\left(\frac{1}{n}\right).$$

From postulate 1 we conclude also that

$$s\left(\frac{1}{n+1}\right) - s\left(\frac{1}{n}\right) \xrightarrow{n \rightarrow \infty} 0,$$

so that lemma 2.1 applies to $s(1/n)$ and hence

$$s(1/n) = K \log n \quad \forall n \in \mathbb{N}.$$

This can be extended to the rational numbers by the additivity of s . For $m, n \in \mathbb{N}$, $m < n$ we have

$$s\left(\frac{m}{n}\right) = s\left(\frac{mn}{mn^2}\right) - s\left(\frac{n}{mn}\right) = s\left(\frac{1}{n}\right) - s\left(\frac{1}{m}\right) = K \log n - K \log m = -K \log \frac{m}{n}.$$

Since the rationals are dense in \mathbb{R} , the continuity of $s(p)$ (postulate 1) immediately gives

$$s(p) = -K \log p, \quad \forall p \in (0, 1].$$

Further, postulate 2 demands $1 = s(1/k) = K \log k = K' \log_k k = K'$, giving $s(p) = -\log_k p$.

Finally, we may write any generalized probability distribution $\mathcal{P} = \{p_1, p_2, \dots, p_n\}$ as $\mathcal{P} = \bigcup_{i=1}^n \mathcal{P}_i$, with $\mathcal{P}_i = \{p_i\}$, and by induction on postulate 3 it is a simple matter to conclude

$$S(\mathcal{P}) = S\left(\bigcup_{i=1}^n \mathcal{P}_i\right) = \frac{\sum_i W(\mathcal{P}_i)S(\mathcal{P}_i)}{\sum_i W(\mathcal{P}_i)} = -\frac{1}{W(\mathcal{P})} \sum_i p_i \log_k p_i,$$

which proves the theorem. \square

Different choices of k are related to each other by multiplication with a constant, so that we may view choosing k as choosing units for $S(X)$. Most common is to use $k = 2$, in which case S is said to be measured in *bits*, or $k = e$, giving S in the units of *nats*.

The Shannon entropy has a set of properties which are easy to derive, but nevertheless helpful to state:

- $S(X) \geq 0$, following from that $p(x) \in [0,1] \forall x$, since this gives $p(x) \log p(x) \leq 0$.
- $S(X) = 0$ iff for some x' , $p(x) = \delta_{x,x'}$ with $\delta_{x,x'}$ being the Kronecker delta.
- $S(X) \leq \log |\mathcal{X}|$, where $|\mathcal{X}|$ is the cardinality of \mathcal{X} .

The third property can be shown by employing Lagrangian multipliers; introduce $S(X, \lambda) \equiv S(X) + \lambda(\sum_x p(x) - 1)$ and set the derivative with respect to $p(x)$ to zero. This gives

$$0 = \frac{\partial S(X, \lambda)}{\partial p(x)} = -\log p(x) - 1 + \lambda \implies p(x) = e^{1-\lambda}, \quad (2.7)$$

which, crucially, is independent of x . Solving the constraint then gives $p(x) = 1/|\mathcal{X}|$ and hence $S(X) = \log |\mathcal{X}|$ as our single local optimum, which is also quite easily seen to be the global maximum.

2.1.1 Joint entropies

If we instead consider a pair of random variables X, Y with joint distribution $p(x, y)$, their joint entropy is naturally defined as

$$S(X, Y) \equiv \mathbb{E}[i(X, Y)] = -\sum_{x, y} p(x, y) \log p(x, y), \quad (2.8)$$

and we may also define the conditional entropy $S(Y|X)$ as the expected information upon observing Y after first observing X , or

$$S(Y|X) \equiv -\sum_x p(x) \sum_y p(y|x) \log p(y|x) = -\sum_{x, y} p(x, y) \log p(y|x). \quad (2.9)$$

Using the above definitions it is easy to see that

$$S(X, Y) = S(X) + S(Y|X) = S(X|Y) + S(Y). \quad (2.10)$$

It is also evident that, if X and Y are independent, i.e. if $p(x, y) = p_X(x)p_Y(y)$, then $S(Y|X) = S(Y)$ and hence $S(X, Y) = S(X) + S(Y)$. Both of these properties makes intuitive sense; the expected information of observing Y should not be affected by first observing X if they are independent, and the total information from independent events should intuitively be the sum of the information from each event.

2.1.2 Relative entropy/Kullback-Liebler divergence

Relative entropy, or Kullback-Liebler (KL) divergence, is a quantifier of the difference between two probability distributions $p(x)$ and $q(x)$ and is defined as [34]

$$D(p \parallel q) \equiv \sum_x p(x) \log \frac{p(x)}{q(x)} = \mathbb{E}_p \left[\log \frac{p(X)}{q(X)} \right]. \quad (2.11)$$

Proposition 2.1. $D(p \parallel q) \geq 0$, with equality only for $p = q$.

Proof. We begin by noting that $\log a \leq a - 1$ for all $a \in \mathbb{R}_+$, with equality only for $a = 1$. This gives

$$D(p \parallel q) = - \sum_x p(x) \log \frac{q(x)}{p(x)} \geq - \sum_{x \in \text{supp}(p)} p(x) \left(\frac{q(x)}{p(x)} - 1 \right) \geq \sum_x p(x) - \sum_x q(x) = 0. \quad (2.12)$$

We may now note that if $q \neq p$ there is some x where $q(x)/p(x) \neq 1$, making the left inequality above strict and finishing the proof. \square

Due to the above inequality, the Kullback-Liebler divergence is often used similarly to a metric on the space of probability distributions, but it is not a metric in the mathematical sense since it both lacks the symmetry between p and q and does not fulfil the triangle inequality.

2.1.3 Fisher information matrix

Although the Kullback-Liebler divergence is not a metric for arbitrary choices of p and q , it approaches one as p and q becomes sufficiently close to each other. To see this, consider a probability distribution $p_\theta(x)$, parametrised by N parameters θ^μ . A series expansion of the KL-divergence between p_θ and a small deviation $p_{\theta+\Delta}$ becomes

$$D(p_\theta \parallel p_{\theta+\Delta}) = -\Delta^\mu \sum_x p_\theta(x) \frac{\partial}{\partial \theta^\mu} \log p_\theta(x) - \Delta^\mu \Delta^\nu \sum_x p_\theta(x) \frac{\partial}{\partial \theta^\mu} \frac{\partial}{\partial \theta^\nu} \log p_\theta(x) + \mathcal{O}(\Delta^3). \quad (2.13)$$

The term linear in Δ is

$$\sum_x p_\theta(x) \frac{\partial}{\partial \theta^\mu} \log p_\theta(x) = \frac{\partial}{\partial \theta^\mu} \sum_x p_\theta(x) = 0, \quad (2.14)$$

where the last step follows from that p_θ is a probability distribution.

We are hence left with

$$D(p_\theta \parallel p_{\theta+\Delta}) = g_{\mu\nu} \Delta^\mu \Delta^\nu + \mathcal{O}(\Delta^3), \quad (2.15)$$

where

$$g_{\mu\nu}(p_\theta) = - \sum_x p_\theta(x) \frac{\partial}{\partial \theta^\mu} \frac{\partial}{\partial \theta^\nu} \log p_\theta(x). \quad (2.16)$$

This $g_{\mu\nu}$ is known as the *Fisher Information Matrix* and can be shown to constitute a metric for the parameter space, defining a Riemannian manifold on which distances are a measure of distinguishability between probability distributions, given data generated from them [35].

2.1.4 Mutual information

The relative entropy may then be used to construct another entity of interest, namely the mutual information between two random variables X and Y . It is a measure of the correlation between the variables and is defined as the KL divergence between the joint probability density $p_{X,Y}(x,y)$ and the product of the marginal distributions $p_X(x)$ and $p_Y(y)$;

$$I(X;Y) \equiv D(p_{X,Y} \| p_X p_Y) = \sum_{x,y} p_{X,Y}(x,y) \log \frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}. \quad (2.17)$$

From the properties of the relative entropy, it is clear that $I(X;Y) \geq 0$ with equality only if X and Y are independent. One can also relate mutual information to the previous entropic measures as

$$\begin{aligned} I(X;Y) &= S(X) + S(Y) - S(X,Y) \\ &= S(Y) - S(Y|X) \\ &= S(X) - S(X|Y). \end{aligned} \quad (2.18)$$

2.2 Rényi entropy

Since its introduction by Shannon, information theoretic entropy has been adapted and generalized in several ways. One of the more notable developments was that by Alfred Rényi[32], who considered the most important properties of a measure of entropy to be the first three postulates of Shannon entropy, together with the additivity for independent random variables. He found that these properties also held if the fourth postulate was relaxed to

$$S_\alpha(\mathcal{P} \cup \mathcal{Q}) = g_\alpha^{-1} \left\{ \frac{W(\mathcal{P})g_\alpha(S(\mathcal{P})) + W(\mathcal{Q})g_\alpha(S(\mathcal{Q}))}{W(\mathcal{P}) + W(\mathcal{Q})} \right\} \quad (2.19)$$

with $g_\alpha(x) \equiv k^{(\alpha-1)x}$ and $\alpha \in [0,1) \cup (1, \infty)$, and that the most general measure satisfying the resulting postulates is the *Rényi entropy*

$$S_\alpha \equiv \frac{1}{1-\alpha} \log_k \left(\sum_i p_i^\alpha \right). \quad (2.20)$$

For $\alpha \rightarrow 1$ the Rényi entropy approaches the Shannon entropy, which is easily seen by invoking l'Hopital's rule;

$$\lim_{\alpha \rightarrow 1} S_\alpha = \lim_{\alpha \rightarrow 1} \frac{1}{1-\alpha} \log \left(\sum_i p_i^\alpha \right) = \lim_{\alpha \rightarrow 1} \frac{\frac{d}{d\alpha} \log \left(\sum_i p_i^\alpha \right)}{\frac{d}{d\alpha} (1-\alpha)}. \quad (2.21)$$

Here the denominator is just -1 , while the numerator is

$$\frac{d}{d\alpha} \log \left(\sum_i p_i^\alpha \right) = \frac{1}{\sum_i p_i^\alpha} \sum_i \frac{d}{d\alpha} p_i^\alpha e^{\alpha \log p_i} = \frac{1}{\sum_i p_i^\alpha} \sum_i p_i^\alpha \log p_i \xrightarrow{\alpha \rightarrow 1} \sum_i p_i \log p_i, \quad (2.22)$$

leaving us with the result

$$\lim_{\alpha \rightarrow 1} S_\alpha = - \sum_i p_i \log p_i, \quad (2.23)$$

which is the definition of Shannon entropy.

The Rényi entropy is also non-increasing in α , which we can show by differentiating. Defining $z_i \equiv p_i^\alpha / \sum_j p_j^\alpha$ and noting that $\sum_i z_i = 1$ we get

$$\begin{aligned}
 \frac{d}{d\alpha} S_\alpha &= \frac{1}{(1-\alpha)^2} \left(\log \sum_i p_i^\alpha + (1-\alpha) \sum_i z_i \log p_i \right) \\
 &= \frac{1}{(1-\alpha)^2} \left(\log \sum_i p_i^\alpha + \sum_i z_i \log \frac{p_i}{z_i \sum_j p_j^\alpha} \right) \\
 &= -\frac{1}{(1-\alpha)^2} \left[\sum_i z_i \log \frac{z_i}{p_i} + \left(\sum_i z_i - 1 \right) \log \sum_j p_j^\alpha \right] \\
 &= -\frac{1}{(1-\alpha)^2} \sum_i z_i \log \frac{z_i}{p_i} = -\frac{1}{(1-\alpha)^2} D(z \| p) \leq 0,
 \end{aligned} \tag{2.24}$$

since the relative entropy is positive semi-definite.

Some of the quantities related to the Shannon entropy which were introduced earlier generalize to the Rényi entropy. Since these generalizations are quite non-trivial, and some of the most useful relations between the quantities turn out to only hold for the Shannon equivalents, it will be informative to also introduce them here.

2.2.1 Rényi divergence

The Rényi divergence generalizes the relative entropy, or KL divergence and is defined as

$$D_\alpha(p \| q) \equiv \frac{1}{\alpha - 1} \log \sum_x p^\alpha(x) q^{1-\alpha}(x). \tag{2.25}$$

It can be shown[36] to fulfil $D_\alpha(p \| q) \geq 0$ and can hence similarly to the relative entropy be used as an estimator of the difference between probability distributions. It is also the case that $\lim_{\alpha \rightarrow 0} D_\alpha(p \| q) = D(p \| q)$, which establishes D_α as a generalization of the relative entropy.

2.2.2 Rényi mutual information

Just as with the Shannon mutual information, the Rényi mutual information is best defined as

$$I_\alpha(X; Y) \equiv D_\alpha(p_{X,Y} \| p_X p_Y). \tag{2.26}$$

An immediate consequence of the nonnegativity of D_α is that also I_α is nonnegative, and it is also easily seen that $\lim_{\alpha \rightarrow 1} I_\alpha(X; Y) = I(X; Y)$ and that $I_\alpha(X; Y) = 0$ if X and Y are statistically independent. We may hence interpret also the Rényi mutual information as quantifying how much knowledge is gained about X by observing Y or vice versa. However, unfortunately there is to the best of our knowledge no equivalent to the relations of equation (2.18), and in particular $I_\alpha(X; Y) \neq S_\alpha(X) + S_\alpha(Y) - S_\alpha(X, Y)$.

3

Quantum information theory

Many of the concepts of information theory have a natural generalisation to the study of quantum many-body systems. In and of itself, this is not surprising since quantum mechanics is fundamentally a probabilistic theory. However, as we will see in the sections to come, quantum mechanics has the peculiarity that one can have full knowledge of the state of a composite system without knowing the states of the parts; or rather, the full system can be in a known state, while subsystems are in superpositions of known states, inducing uncertainty in any measurement on the subsystems in question. In addition, we may also have an uncertainty as to the full state of the system, which is akin to what one expects of a classical description. These properties make the application of information theory to quantum mechanics a very interesting endeavour, with quite a few unexpected quirks along the way. Much of this chapter largely follows the developments of [29], which I highly recommend for a further reading. Another, much more detailed, resource on the subject is [30].

We will begin this chapter with a very brief introduction of the mathematical constructs involved and the Dirac notation which is commonly used throughout the subject. A reader who has come in contact with quantum mechanics before may well wish to skip the first few sections.

3.1 Quantum theory – the bare minimum

The fundamental description of the state of a quantum mechanical system is in terms of a vector in a Hilbert space \mathcal{H} , often written on the form

$$|\psi\rangle \in \mathcal{H}. \quad (3.1)$$

Elements of the dual space \mathcal{H}^\dagger is written as $\langle\varphi|$, and the inner product $\langle\varphi|\psi\rangle$. Note that the corresponding notation in the mathematics community would be $\langle\psi, \varphi\rangle$ with the vectors written instead as ψ and φ . A quantum state has the interpretation of a probability amplitude, and is hence most often normalized to $\langle\psi|\psi\rangle = 1$.

We interact with a quantum state through making measurements of observables, both of which we will now define.

An observable is a Hermitian operator $O \in \text{end}(\mathcal{H})$. Owing to its Hermiticity, it has $n = \dim \mathcal{H}$ (possibly degenerate) real eigenvalues λ_i , $i = 1, \dots, n$ with corresponding eigenvectors $|i\rangle$. These eigenvectors can be constructed to form an orthonormal basis over \mathcal{H} , meaning that any state $|\psi\rangle$ can be written on the form

$$|\psi\rangle = \sum_i c_i |i\rangle, \quad \sum_i |c_i|^2 = 1, \quad (3.2)$$

with $c_i \in \mathbb{C}$.

A measurement¹ of the non-degenerate observable O is then most easily thought of as a random process, where one of the eigenvalues λ_i is measured with probability

$$p(O = \lambda_i) = |\langle i|\psi\rangle|^2 = |c_i|^2, \quad (3.3)$$

and the state $|\psi\rangle$ is projected onto the corresponding eigenvector $|i\rangle$. The case of a degenerate observable is similar, although the projection is then onto the entire eigenspace of the eigenvalue.

We are often interested in the expectation value of an observable, which by the above procedure may be expressed as

$$\langle O \rangle \equiv \langle \psi|O|\psi\rangle = \sum_i \lambda_i |c_i|^2. \quad (3.4)$$

A particularly interesting measurable quantity is the energy E of a system. Its corresponding observable is called the Hamilton operator H and governs the time development of a state by the Schrödinger equation

$$i\frac{\partial}{\partial t} |\psi\rangle = H |\psi\rangle. \quad (3.5)$$

There are remarkably few Hamiltonians for which the eigenspectrum can be found analytically, and finding the spectra for those that remain is a major issue in numerical quantum mechanics, which we will deal with in more detail in chapter 6. This is especially the case for systems of many particles, or many-body systems, since their state spaces have an exponential dependence upon the number of particles.

3.2 Composite quantum mechanical systems

A central feature where quantum mechanics deviates significantly from our classical intuition is description of composite systems. Classically we are used to being able to characterise the state of all subsystems independently; You do not need to know the entire state of the universe to know the position of your left hand. To put it more concretely, consider a system of interest A (e.g. your left hand) and a system B containing all other things of relevance (possibly the rest of the universe), and let the space of all possible states for either system be \mathcal{A} and \mathcal{B} respectively. Then, classically (although this formalization is rather unusual), one can think of the state space of the composite system as $\mathcal{AB} = \mathcal{A} \oplus \mathcal{B}$, so that the full state can be specified by choosing an element in \mathcal{AB} , which is the same as independently choosing elements from \mathcal{A} and \mathcal{B} . Hence, no knowledge of B is needed to fully characterise a state in A .

In the realm of quantum mechanics however, we lack the luxury of locality. Here, the individual states of systems A and B , characterised as vectors $|\varphi_A\rangle$ and $|\varphi_B\rangle$ in Hilbert spaces \mathcal{H}_A and \mathcal{H}_B , instead form a composite state space $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$ and the full description of the state is achieved by choosing a vector $|\psi\rangle \in \mathcal{H}$. Now, had $|\psi\rangle$ been on the form $|\psi\rangle = |\varphi_A\rangle \otimes |\varphi_B\rangle$, we would have our full description formed by independently choosing states from A and B in line with our classical reasoning. However, not all vectors in \mathcal{H} can be put on this form. To see this, we introduce orthonormal bases $|i_A\rangle$ and $|i_B\rangle$

¹This view of the measurement process is known as the Copenhagen interpretation, and is only one of many ways of interpreting the seemingly stochastic nature of quantum mechanics. Other interpretations include the many-worlds interpretation, pilot waves and qubism. Since they all lead to more or less the same mathematics and the philosophical differences, while very interesting, are hardly relevant for this text, we might as well choose the simplest one and proceed.

on \mathcal{H}_A and \mathcal{H}_B , where $i_X = 1, \dots, \dim(\mathcal{H}_X)$. The set of all products $|i_A\rangle \otimes |j_B\rangle$ then forms a basis on \mathcal{H} , giving the description of a generic state in this space as

$$|\psi\rangle = \sum_{i,j} c_{ij} |i_A\rangle \otimes |j_B\rangle, \quad (3.6)$$

where the $c_{ij} \in \mathbb{C}$ are commonly normalised to $\sum_{i,j} |c_{ij}|^2 = 1$. The state can be written on the product form $|\varphi_A\rangle \otimes |\varphi_B\rangle$ iff there exists vectors $(\varphi_A)_i, (\varphi_B)_j$ such that $c_{ij} = (\varphi_A)_i (\varphi_B)_j$ and is then called *separable*. Otherwise, the state is said to be *entangled*, and the subsystems then cannot be fully described independently.

3.3 Schmidt decomposition

As with any matrix, we can make a singular value decomposition (SVD) of c_{ij} , as $c_{ij} = \sum_{k,l} U_{ik} c'_{kl} V_{lj}$, where U and V are unitary operators on \mathcal{H}_A and \mathcal{H}_B respectively and $c'_{kl} = \lambda_k \delta_{kl}$, with λ_k being the singular values, which can be chosen real and non-negative. In fact, this amounts to a unitary change of basis,

$$|\psi\rangle = \sum_{i,j,k,l} c'_{kl} U_{ik} V_{lj} |i_A\rangle \otimes |j_B\rangle = \sum_{i=1}^d \lambda_i |i'_A\rangle \otimes |i'_B\rangle \quad (3.7)$$

where $d \leq \min(|\mathcal{H}_A|, |\mathcal{H}_B|)$ is the number of non-zero eigenvalues, and $|i'_A\rangle = \sum_j U_{ji} |j_A\rangle$, and $|i'_B\rangle = \sum_j V_{ij} |j_B\rangle$ form the new orthonormal bases for \mathcal{H}_A and \mathcal{H}_B respectively. We may now interpret $\lambda_i^2 =: p_i$ as the probability of finding the system in state $|i'_A i'_B\rangle = |i'_A\rangle \otimes |i'_B\rangle$, and write, dropping the primes for convenience,

$$|\psi\rangle = \sum_{i=1}^d \sqrt{p_i} |i_A i_B\rangle. \quad (3.8)$$

This construction is called a *Schmidt decomposition*, while d is called the *Schmidt number*. To interpret the Schmidt number we now consider the expectation value of any observable O in system A . In the composite system we may write this as $O \otimes \text{id}_B$, giving the expectation value as

$$\langle O \rangle_A \equiv \langle \psi | O \otimes \text{id}_B | \psi \rangle = \sum_{i,j} \sqrt{p_i p_j} \langle i_A | O | j_A \rangle \langle i_B | \text{id}_B | j_B \rangle = \sum_{i=1}^d p_i \langle i_A | O | i_A \rangle. \quad (3.9)$$

From this we see that the number of terms in the sum is equal to the Schmidt number, giving d as a crude measure of the uncertainty induced from the state being in a composite Hilbert space, although this uncertainty would of course also be dependent on the individual probabilities p_i .

The common term for this “induced uncertainty” is *entanglement* between A and B , of which, as noted, the Schmidt number is a crude estimator. As the title of this chapter hints, we shall soon be able to introduce a few finer measures of entanglement, which are more or less directly translated from information theory. First, however, we need a way of describing entangled states without reference to the entire system.

3.4 The density matrix

Since we do not in general possess full knowledge of the state of the universe, it is desirable to achieve a way of calculating expectation values without reference to the subsystem B .

Looking at (3.9) we see that

$$\langle O \rangle_A = \sum_i p_i \langle i_A | O | i_A \rangle = \text{tr}_{\mathcal{H}_A} \left[\sum_i \langle i_A | O p_i | i_A \rangle \right] = \text{tr}_{\mathcal{H}_A} \left[O \sum_i p_i | i_A \rangle \langle i_A | \right], \quad (3.10)$$

where we have used the linearity and the cyclicity of the trace as well as the linearity of O . This allows us to introduce the *the density matrix*

$$\rho_A \equiv \sum_i p_i | i_A \rangle \langle i_A |, \quad (3.11)$$

giving the expectation value of O as

$$\langle O \rangle_A = \text{tr} [O \rho_A]. \quad (3.12)$$

which both is agnostic of B and basis invariant; under a unitary change of basis we find

$$\text{tr} [O \rho_A] \rightarrow \text{tr} [U^\dagger O U U^\dagger \rho_A U] = \text{tr} [O \rho_A U U^\dagger] = \text{tr} [O \rho_A]. \quad (3.13)$$

We should also note that we may arrive at ρ_A from the full state $|\psi\rangle$ in a very simple manner. Consider first the density matrix for the full system, which for a pure state is defined as

$$\rho \equiv |\psi\rangle \langle \psi|, \quad (3.14)$$

and in the basis of the Schmidt decomposition becomes

$$\rho = \sum_{i,j} \sqrt{p_i p_j} | i_A i_B \rangle \langle j_A j_B |. \quad (3.15)$$

Now, to retrieve the reduced density matrix ρ_A , we simply perform a partial trace over \mathcal{H}_B ;

$$\begin{aligned} \text{tr}_{\mathcal{H}_B} \rho &= \text{tr}_{\mathcal{H}_B} \sum_{i,j} \sqrt{p_i p_j} | i_A i_B \rangle \langle j_A j_B | = \sum_{i,j} \sqrt{p_i p_j} | i_A \rangle \langle j_A | \text{tr}_{\mathcal{H}_B} [| i_B \rangle \langle j_B |] \\ &= \sum_i p_i | i_A \rangle \langle i_A | = \rho_A. \end{aligned} \quad (3.16)$$

Some properties that we may note from the form of the density matrix is that it is (i) Hermitian, (ii) positive semi-definite and (iii) has trace one, $\text{tr}_{\mathcal{H}_A} \rho_A = \sum_i p_i = 1$.

In the above we have only considered a pure composite state where we know the Schmidt decomposition, but it is a simple matter to extend this also to non-pure, or *mixed* states. These are states where the composite system is in a mixture of several pure states $|\psi^i\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$, each with weight $\lambda_i > 0$. We may then write the density matrix, which is still Hermitian and positive semi-definite, as

$$\rho \equiv \sum_i \lambda_i |\psi^i\rangle \langle \psi^i|, \quad (3.17)$$

with the λ_i normalized such that $\text{tr} \rho = 1$. In a generic orthonormal basis this becomes, for some tensor c_{ijkl} ,

$$\rho = \sum_{i,j,k,l} c_{ijkl} | i_A j_B \rangle \langle k_A l_B |. \quad (3.18)$$

The reduced density matrices are, just like before, retrieved by tracing out the other system,

$$\rho_A = \text{tr}_{\mathcal{H}_B} \rho = \sum_{i,j,k} c_{ikjk} | i_A \rangle \langle j_A |, \quad (3.19)$$

allowing single-system operators to be calculated without explicit knowledge of the other system.

An interesting aside regarding density matrices is that of *purification*. Consider a generic density matrix $\rho_A \in \text{hom}(\mathcal{H}_A)$. Since it is Hermitian, we can always find an orthonormal basis of \mathcal{H}_A where it is diagonal. In this basis,

$$\rho_A = \sum_{i=1}^d p_i |i_A\rangle \langle i_A|, \quad (3.20)$$

for some $d \leq \dim H_A$, and $p_i \in \mathbb{R}$. Now introduce an artificial Hilbert space \mathcal{H}_B with $\dim H_B \geq d$ and ON-basis $|i_B\rangle$ and let

$$\rho \equiv |\psi\rangle \langle \psi|, \quad |\psi\rangle \equiv \sum_{i=1}^d \sqrt{p_i} |i_A\rangle \otimes |i_B\rangle. \quad (3.21)$$

Then, as we have already seen, $\rho_A = \text{tr}_{\mathcal{H}_B} \rho$. Since $|\psi\rangle$ is a pure state, we call it a purification of ρ_A .

3.5 Measures of entanglement

Having properly introduced the relevant parts of the density matrix formalism we are now ready to take a proper look at some different ways to quantify the uncertainty and/or entanglement in a quantum many-body state. The measures we will consider are the Schmidt number (which we have already introduced), the von Neumann entropy, which is a generalization of the Shannon entropy, and the quantum Rényi entropy analogously extended from the classical Rényi entropy.

3.5.1 Schmidt number

The simplest measure of entanglement is the Schmidt number, which we have already defined in the context of pure states as the number of terms in the Schmidt decomposition. To extend this to generic density matrices, we note that the number of terms in the purification of a state (see eq. (3.21)) is equal to the number of non-zero eigenvalues of the original density matrix ρ . But this is equal to the matrix rank of ρ , so that we may write the Schmidt number as

$$\text{rank } \rho. \quad (3.22)$$

As we expect for a measure of entanglement, this is minimal for a pure state, which has a density matrix with a single non-zero eigenvalue, and maximal for a density matrix which assigns non-zero probability to all states in its Hilbert space.

3.5.2 von Neumann entropy

As first noted by von Neumann, Shannon entropy (or equivalently, Gibbs entropy) may be quite naturally extended to the domain of quantum statistical mechanics. Also here we may interpret it as the uncertainty in the state, or the expected information retrieved from making a measurement. Consider a density matrix ρ , which we may diagonalize as $\rho = \sum_i p_i |\psi_i\rangle \langle \psi_i|$, where the $|\psi_i\rangle$ are orthonormal. Since the p_i are ≥ 0 and sum to one, a natural interpretation is that, if we make a measurement whose operator is diagonal in the

same basis, we will find the system to be in state $|\psi_i\rangle$ with probability p_i . The expected uncertainty of this measurement is then the Shannon entropy

$$S(\rho) = - \sum_i p_i \log p_i, \quad (3.23)$$

or, to put it in a slightly more invariant form,

$$S(\rho) \equiv - \text{tr}[\rho \log \rho]. \quad (3.24)$$

This is known as the *von Neumann entropy* of the density matrix ρ . From eq. (3.23) we see that, as with Shannon entropy,

$$S(\rho) \geq 0, \quad (3.25)$$

with equality only for pure states, i.e. when $p_i = \delta_{ij}$ for some j .

3.5.3 Quantum Rényi entropy

The generalization of Shannon entropy to Rényi entropy carries over to the quantum case in the simplest possible manner, namely

$$S_q(\rho) \equiv \frac{1}{1-q} \log \text{tr}(\rho^q). \quad (3.26)$$

In fact, diagonalizing ρ , so that $\rho = \text{diag}\{p_1, p_2, \dots\}$, with p_i being the eigenvalues of ρ we see that this reduces to the classical definition, just as in the case of von Neumann entropy, allowing us to conclude that

$$\lim_{q \rightarrow 1} S_q(\rho) = S(\rho), \quad (3.27)$$

as well as

$$\frac{d}{dq} S_q(\rho) \leq 0 \quad (3.28)$$

holds also in the quantum case.

One aspect of the Rényi entropy that is easily overlooked is its computational simplicity – since the trace is taken before the logarithm, and integer powers of matrices can be computed without series expansion or knowledge of the eigenvalues, $S_q(\rho)$ becomes a much simpler function of ρ for integer $q \neq 1$ than $S(\rho)$. In particular, even if we are only interested in von Neumann entropy, we can use the monotonicity of Rényi entropy to bound it as

$$S_2(\rho) \leq S(\rho) \leq S_0(\rho). \quad (3.29)$$

Here the zeroth Rényi entropy (also known as Hartley entropy) is particularly simple to calculate; if we again consider a diagonalized density matrix $\rho = \text{diag}\{p_1, p_2, \dots\}$ and define $0^0 \equiv 0$ for convenience (and for continuity at $q = 0$), we find

$$S_0(\rho) = \log \text{tr} \rho^0 = \log \text{tr} \sum_i p_i^0 |i\rangle \langle i| = \log \text{rank } \rho, \quad (3.30)$$

which we note is just the logarithm of the Schmidt number.

3.5.4 Equal entropy of subsystems

An interesting consequence of the Schmidt decomposition is that the entropies of the subsystems of a pure bipartite state are equal. To see this, let $|\psi\rangle = \sum_i \sqrt{p_i} |i_A i_B\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$ be such a state, and let $\rho = |\psi\rangle\langle\psi|$ be its density matrix. Then,

$$\rho_A = \text{tr}_{\mathcal{H}_B} \rho = \sum_i p_i |i_A\rangle\langle i_A|, \quad \rho_B = \text{tr}_{\mathcal{H}_A} \rho = \sum_i p_i |i_B\rangle\langle i_B|, \quad (3.31)$$

so that

$$S_q(A) = S_q(\{p_i\}) = S_q(B), \quad (3.32)$$

where we have introduced $S_q(X) \equiv S_q(\rho_X)$ to simplify notation.

3.5.5 Quantum mutual information

Not all of the entropy-related quantities introduced in the previous chapter generalize well to the quantum entropies, but one that do is the mutual information. Consider a bipartite quantum mechanical system AB with density matrix $\rho_{AB} \in \text{end}(\mathcal{H}_A \otimes \mathcal{H}_B)$. Then the (quantum) mutual information between A and B is defined as

$$I(A : B) \equiv S(A) + S(B) - S(AB), \quad (3.33)$$

where $S(X)$ is the von Neumann entropy of system X . It is non-negative and may readily be interpreted as the amount of information obtained about A by making a measurement on B [29].

A useful property of this mutual information is that, for a pure state $\rho_{AB} = |\psi\rangle\langle\psi|$, we know the joint entropy to be $S(AB) = 0$, while the reduced entropies are equal, $S(A) = S(B)$, giving

$$I(A : B) \Big|_{\rho_{AB} \text{ is pure}} = 2S(A) = 2S(B), \quad (3.34)$$

so that the entropy of either of the reduced regions itself in this case becomes a measurement also of the mutual information, or correlation, between A and B .

4

Graph theory

Graph theory is a powerful tool both in the context of machine learning, where graphs are often used to represent probability distributions with complex correlation structures (see e.g. [37, ch.16]) and in quantum many-body theory where they form the basis of Tensor Networks, which is a tool for describing quantum states with very general entanglement properties [38]. Both of these uses will be relevant to our later developments, and it is therefore helpful to go through some useful definitions and one or two results from the subject.

4.1 Types of graphs

One of the standard definitions of a graph is the following:

Definition 4.1. (Graph) A graph $G = (V, E)$ is an ordered pair of disjoint sets V and E , where $E \subseteq [V]^2$, with $[V]^2$ being the set of unordered pairs $\{v, w\}$ of elements $v, w \in V$. The elements $v \in V$ are called *vertices* or *nodes* and the elements $e \in E$ *edges*.

As the name implies, graphs lend themselves well to visual representations. Typically one draws each vertex as a dot and each edge as a line between the corresponding vertices, as shown in fig 4.1a.

However, in the context of tensor networks, we will need to also allow for edges that are not connected to two vertices. Such graphs, often called *open* or *half-edge* graphs seem to rarely be formalised, perhaps since they are so easily translated to the standard, or *closed* graphs. Nevertheless, inspired by the definition in [39] we define them as follows.

Definition 4.2. (Open Graph) An open graph $G = (V, \varepsilon, E)$ is an ordered triplet of disjoint sets V , ε and E , with $E \subseteq [V \cup \varepsilon]^2$ such that each element in ε is only contained in one element of E . The elements $x \in \varepsilon$ are called *edge points*, while, as before, elements $v \in V$ are called vertices and the elements $e \in E$ edges.

We will often make ε implicit and refer also to open graphs as $G = (V, E)$. In these cases it should be clear from the context what is meant. A visual depiction of an open graph is shown in fig. 4.1b.

There will sometimes be a need to distinguish *internal* edges $e \in \bar{E} \equiv E \cap [V]^2$, which are connected to two vertices, from *dangling* edges $e \in \partial E \equiv E \setminus [V]^2$, which have one or both ends “free”. For this we will use the above introduced notations \bar{E} and ∂E . In addition, we will often write the set of edges connected to a vertex $v \in V$ as $dv \equiv \{e \in E; e \ni v\}$, and the set of vertices connected to an edge $e \in E$ as $\partial e \equiv V \cap e$.

There is a wealth of standard literature dealing with closed graphs. Since this is, to the best of our knowledge, less true for open graphs it is helpful to be able to express an

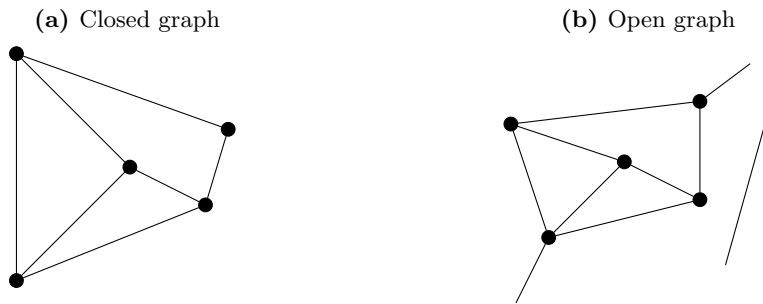


Figure 4.1: Visual representations of closed and open graphs respectively. Each point corresponds to a vertex and each line to an edge.

open graph in terms of a closed. This we can easily do by making the following definition, inspired by [40].

Definition 4.3. The completion of the open graph $G = (V, \varepsilon, E)$ is the closed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = V \cup \varepsilon$ and $\mathcal{E} = E$.

When doing graph theory on open graphs in later chapters, we will almost exclusively be formally dealing with the completion of the graph.

It will also be helpful to introduce the following definitions:

A *multigraph* is a graph $G = (V, E)$ where we allow several edges between the same set of vertices, i.e. for $e_1, e_2 \in E$ we allow both e_1 and e_2 to correspond to the pair $\{v, w\} \in [V]^2$. Formally this can be achieved e.g. by constructing the correspondence between E and $[V]^2$ as an incidence function $\psi_G : E \rightarrow [V]^2$. However we will still, with some slight notational abuse, refer to edges as $e = \{v, w\}$. An open multigraph is defined analogously.

A *directed* graph is a graph $G = (V, E)$ where the edges $e \in E$ are ordered pairs of vertices, $e = (v, w)$, with $v, w \in V$. Visually, an edge $e = (v, w)$ is represented as an arrow from v to w .

A *weighted* graph is a graph $G = (V, E)$ together with a weight function $w : E \rightarrow \mathbb{R}_+$.

4.2 Paths, cuts and cliques

A *trail* is a finite alternating sequence of vertices and distinct edges $v_1 e_1, v_2 e_2, \dots$ such that $e_i = (v_i, v_{i+1})$.

A *path* is a trail where also the vertices are distinct.

A *circuit* is a trail that starts and ends at the same vertex.

A graph is said to be *connected* if there is a path between any two vertices in the graph.

An *edge cut set* of a connected graph $G = (V, E)$ is a set of edges $C \subseteq E$ such that the graph $G' = (V, E \setminus C)$ is disconnected. It is said to separate $A \subset V$ from $B \subset V$ if there is no path between A and B in G' . The weight of a cut C is given by

$$|C| \equiv \sum_{e \in C} w(e), \quad (4.1)$$

where $w(e)$ is the weight function of the graph. For unweighted graphs one typically takes $w(e) \equiv 1$, in which case $|C|$ becomes the cardinality of C .

Definition 4.4. (Min cut)

Let \mathcal{C} be the family of all edge cut sets in the graph G . The min cut of the graph is then $\text{MC}(G) \equiv \min_{C \in \mathcal{C}} |C|$.

Let $\mathcal{C}_{(A,B)}$ be the family of all edge cut sets in the graph G separating A from B . The min cut separating A from B is then $\text{MC}_{(A,B)}(G) \equiv \min_{C \in \mathcal{C}_{(A,B)}} |C|$.

Definition 4.5. A clique is a subset of the vertices $C \subseteq V$ such that $(u,v) \in E$ for all $u, v \in C$, or equivalently, such that the subgraph induced by C is complete.

It is *maximal* if it is not a proper subset of another clique.

4.3 Max flow/min cut and Menger's theorem

One of the most interesting and widely applicable sub-areas of graph theory is that of network theory, where one studies flows, which we may think of as consisting of e.g. water, electric current or cars, through graphs which correspondingly may be thought of as networks of water pipes, wires or roads. A central result from network theory, which we will now proceed to discuss, is that the maximum flow through a network may be related to a min cut in the graph. This is known as the max flow/min cut theorem.

Let $G = (V, E, c)$ be a *flow network*, i.e. a directed, weighted graph with weight function (capacity) c and let $S, T \subset V$ be the set of sources (vertices with only outgoing edges) and sinks (vertices with only incoming edges) respectively.

Each edge $e \in E$ is also associated with a flow $f : E \rightarrow \mathbb{R}_+$ which is capped by the capacity, $f(e) \leq c(e)$, and conserved, meaning that for all $v \in V$

$$\sum_{e \in d_{\text{in}}v} f(e) = \sum_{e \in d_{\text{out}}v} f(e), \quad (4.2)$$

where

$$\begin{aligned} d_{\text{in}}v &= \{e \in E; e = (u, v), \text{ for some } u \in V\}, \quad \text{and} \\ d_{\text{out}}v &= \{e \in E; e = (v, u), \text{ for some } u \in V\}. \end{aligned}$$

The *network flow* $|f|$ is the total flow leaving the source (or, equivalently, entering the sink), i.e. $|f| \equiv \sum_{v \in S} \sum_{e \in d_{\text{out}}v} f(e)$. The max flow is defined as $\text{MF}(G) \equiv \max_f |f|$. Note that, for unweighted graphs upon setting $c(e) = 1$, $\text{MF}(G)$ becomes the number of edge-disjoint paths from S to T .

An $S - T$ *cut* is as a set of edges $C_{(S-T)} \subset E$ such that the graph $G' = (V, E \setminus C_{(S-T)})$ contains no paths from S to T . The min $S - T$ cut weight is then defined as $\text{MC}_{(S-T)}(G) = \min_{C \in \mathcal{C}_{(S-T)}} |C|$, with $\mathcal{C}_{(S-T)}$ being the family of all $S - T$ cuts.

Theorem 4.1. [41, 42] (Max flow/min cut theorem)

Let G, S, T be as above. Then $\text{MF}(G) = \text{MC}_{(S-T)}(G)$.

Proof. See e.g. [25]. □

Theorem 4.2. (Menger's Theorem) Let $G = (V, E)$ be a directed, unweighted graph with sources S and sinks T . Then the maximum number of edge-disjoint paths from S to T is equal to the minimal $S - T$ cut weight $\text{MC}_{(S-T)}(G)$.

Proof. Follows immediately from the Max flow/Min cut theorem after setting $c(e) = 1 \forall e \in E$ and noting that the number of edge disjoint paths from S to T then becomes equal to $\text{MF}(G)$. □

Theorem 4.3. [25] (Undirected Menger's theorem)

Let $G = (V, E)$ be an undirected (unweighted) graph, and (A, B) a partition of the vertices with degree 1. Let $\text{MF}_{(A,B)}(G)$ be the maximum number of edge-disjoint paths between A and B . Then $\text{MC}_{(A,B)}(G) = \text{MF}_{(A,B)}(G)$.

Proof. Consider an edge cut set C separating A from B . By definition, any path from A to B must then traverse an edge in C . Since the paths in $\text{MF}_{(A,B)}(G)$ are edge disjoint, there can therefore be a maximum of $|C|$ such paths, giving $\text{MF}_{(A,B)}(G) \leq \text{MC}_{(A,B)}(G)$.

Now, let $G' = (V, E')$ be a directed graph with the same set of vertices as G , but where each undirected edge $(u, v) \in E$ is replaced with the two directed edges (u, v) and (v, u) , unless $u \in A$ or $v \in B$, in which case $(u, v) \in E'$ while $(v, u) \notin E'$, so that A is a source and B a sink for G' . It is clear that $\text{MC}_{(A-B)}(G') = \text{MC}_{(A,B)}(G)$ since any cut set $C \in E$ can be made into a cut set $C' \in E'$ by for each edge $e \in C$ picking a corresponding $e' \in C'$ between the same vertices, chosen in the appropriate direction. By the directed Menger's theorem we have that $\text{MC}_{(A-B)}(G') = \text{MF}_{(A,B)}(G')$.

Given a set of $\text{MF}(G')$ edge-disjoint paths from A to B in G' , it is easily seen that it is always possible to make them non-intersecting and acyclic without reducing the number of paths by swapping edges between intersecting paths and removing the edges that form cycles. Now, consider such a set of $\text{MF}(G')$ non-intersecting and acyclic paths from A to B . It is clear that only one of the edges in G' corresponding to a given edge in G can be traversed by this set of paths, since the paths would have either cycles or intersections otherwise. Hence this set of paths is also a valid set of edge-disjoint paths in G , showing $\text{MF}_{(A,B)}(G) \geq \text{MF}(G') = \text{MC}_{(A,B)}(G)$.

Since we have shown both \leq and \geq , we are left with $\text{MF}_{(A,B)}(G) = \text{MC}_{(A,B)}(G)$, which is the statement of the theorem. \square

4.4 Vertex limited flows

A simple generalization of max flow/min cut, which we will be able to make good use of in the next chapter, is to introduce capacities also to vertices. That is, we extend c to be $c : V \cup E \rightarrow \mathbb{R}_+ \cup \infty$ and demand in addition to before, that the flow

$$f(v) \equiv \sum_{e \in \text{d}_{\text{in}} v} f(e) \leq c(v). \quad (4.3)$$

It will also be beneficial to introduce the notion of combined cuts.

Definition 4.6. A set $C = C_V \cup C_E$, with $C_V \subset V$ and $C_E \subset E$ of a connected graph G is a combined cut set if $G' = (V \setminus C_V, E \setminus C_E)$ is disconnected.

Cut weights and minimal cuts are defined analogously to the edge cut case, bearing in mind that also vertex capacities are to be considered.

Now we will construct a similar series of max flow/min cut theorems for the vertex limited flows, starting with general flows.

Theorem 4.4. (Vertex limited max flow/min cut theorem) Let $G = (V, E)$ be a vertex-limited flow network, with sources S , sinks T and capacities c . Then $\text{MF}(G, c) = \text{MC}(G, c)$.

Proof. We may construct from G an ordinary flow network $G' = (V', E')$ with capacity function $c' : E' \rightarrow \mathbb{R}_+ \cup \infty$, by splitting each vertex $v \in V$ into two vertices v_{in} and v_{out}

in V' , which we connect by an edge $(v_{\text{in}}, v_{\text{out}})$ with capacity $c(v)$. In addition, we connect the edges dv_{in} to v_{in} and dv_{out} to v_{out} .

It is obvious that by this construction we have $\text{MF}(G, c) = \text{MF}(G', c')$ and $\text{MC}(G, c) = \text{MC}(G', c')$, where the left MC is a minimal combined cut weight while the right MC is a minimal edge cut weight. But by the min cut/max flow theorem, $\text{MC}(G', c') = \text{MF}(G', c')$, and the theorem follows. \square

The two above versions of Menger's theorem are also easily adapted:

Theorem 4.5. (Vertex limited Menger's theorem)

Let $G = (V, E)$ be a directed, unweighted graph with sources S and sinks T . Then the maximum number of edge-disjoint paths from S to T such that a maximum of $n_v \in \mathbb{Z}_+$ paths passes through vertex v is equal to the minimal combined cut weight $\text{MC}(G, c)$, with capacity function

$$c(x) = \begin{cases} n_x & \text{if } x \in V, \\ 1 & \text{if } x \in E. \end{cases} \quad (4.4)$$

Proof. Follows with the above choice of $c(x)$ immediately from the vertex limited max flow/min cut theorem after noting that the number of vertex-limited edge disjoint paths from S to T becomes equal to $\text{MF}(G)$. \square

Theorem 4.6. (Vertex limited undirected Menger's theorem)

Let $G = (V, E)$ be an undirected, unweighted graph with sources S and sinks T . Then the maximum number of edge-disjoint paths from S to T such that a maximum of $n_v \in \mathbb{Z}_+$ paths passes through vertex v is equal to the minimal combined cut weight $\text{MC}(G, c)$, with capacity function

$$c(x) = \begin{cases} n_x & \text{if } x \in V, \\ 1 & \text{if } x \in E. \end{cases} \quad (4.5)$$

Proof. The proof is analogous to that of the ordinary undirected Menger's theorem. \square

Of these three theorems, it is mainly the latter one that is of interest to us, as we will be able to use it for studying ranks of matricizations of tensor networks in the next chapter.

5

Tensor networks

One of the biggest obstacles when performing numerical simulations of many-particle quantum mechanical systems is the curse of dimensionality, stemming from that the number of degrees of freedom of a generic many-body state is exponential in the number of particles. However, it seems that physical systems rarely exploit the entire available state space, but rather contain additional structure which allow them to be described with much fewer variables (this phenomenon is also common to the context of ML where it is often referred to as “the blessing of non-uniformity”).

One tool that has proven very useful in creating efficient descriptions of quantum states, especially for condensed matter systems, but also, somewhat surprisingly, in the context of the holographic principle from quantum gravity (see e.g. [43]), is that of Tensor Networks (TN), where one represents a many-body quantum state as a high-dimensional tensor which is decomposed into inner products between several lower dimensional tensors. These inner products induce correlations between the subsystems involved and their structure can easily be represented on a graph, giving an intuitive way to grasp the approximate correlation structure of the state.

This section is an introduction on the subject and to a large degree a summary of [38].

5.1 An introduction

A tensor network is a graphical representation of a multilinear tensor T , which for our purposes is best viewed as an element in some finite-dimensional Hilbert space \mathcal{H} . The tensor is represented on an open multigraph $G = (V, E)$ by assigning to each edge $e \in E$ a finite-dimensional Hilbert space \mathcal{H}^e and to each vertex $v \in V$ a tensor

$$T_v \in \mathcal{H}_v \equiv \bigotimes_{e \in dv} \mathcal{H}^e. \quad (5.1)$$

Note that the number of indices, or modes of T_v becomes equal to the degree $|dv|$ of v . Also, for convenience we will call the total set of tensors in the TN

$$\mathcal{T} \equiv \bigcup_{v \in V} T_v. \quad (5.2)$$

The overall tensor T is then arrived at by for each doubly connected edge $\{u, v\} \in \bar{E}$ taking the inner product between the corresponding modes in T_u and T_v . This results in T being an element of

$$\mathcal{H} = \bigotimes_{e \in \partial E} \mathcal{H}^e, \quad (5.3)$$

where ∂E is the set of dangling edges in G .

The direction of the edges are sometimes (but not always) given special meaning. One such scheme which we will make some use of is to divide the edges of a vertex into two set; $dv = dv_{\text{in}} \cup dv_{\text{out}}$, and view the corresponding tensor as a linear map

$$T_v \in \text{hom}(\mathcal{H}_{\text{in}}, \mathcal{H}_{\text{out}}), \quad \begin{cases} \mathcal{H}_{\text{in}} \equiv \bigotimes_{e \in dv_{\text{in}}} \mathcal{H}^e \\ \mathcal{H}_{\text{out}} \equiv \bigotimes_{e \in dv_{\text{out}}} \mathcal{H}^e. \end{cases} \quad (5.4)$$

The edges in dv_{in} are then drawn pointing downward and those in dv_{out} pointing upward. We will often use upper and lower indices of tensors correspondingly, meaning that the tensor $T^{i_1, \dots, i_m}_{j_1, \dots, j_n}$, with $m = |dv_{\text{out}}|$ and $n = |dv_{\text{in}}|$, may be drawn as

$$T^{i_1, \dots, i_m}_{j_1, \dots, j_n} = \begin{array}{c} i_1 \quad i_2 \quad \dots \quad i_m \\ \diagdown \quad \diagup \quad \dots \quad \diagdown \\ \textcircled{T} \\ \diagup \quad \diagdown \quad \dots \quad \diagup \\ j_1 \quad j_2 \quad \dots \quad j_n \end{array}. \quad (5.5)$$

Note that by this prescription, the dual tensor $T^\dagger \in \text{hom}(\mathcal{H}_{\text{out}}, \mathcal{H}_{\text{in}})$ is to be vertically flipped compared to T . As there is no conceptual difference between edges viewed as inputs and edges viewed as outputs, this distinction is not always needed and we will hence allow ourselves to raise and lower indices and change direction of edges freely. It will often be convenient to draw edges horizontally to save space.

A very simple, yet instructive, example of a non-trivial TN is the matrix product

$$\begin{array}{c} i \\ | \\ \textcircled{T} \\ | \\ \textcircled{V} \\ | \\ j \end{array} \equiv T^i_k V^k_j, \quad (5.6)$$

where we have used the Einstein summation convention for the inner product between T^i_k and V^k_j . Two other examples which should be considered are the identity map and the outer product,

$$\text{id} = \left| \begin{array}{c} | \\ | \\ | \\ | \\ | \end{array} \right|, \quad \text{and} \quad u \otimes v = \begin{array}{c} | \\ \textcircled{u} \\ | \end{array} \quad \begin{array}{c} | \\ \textcircled{v} \\ | \end{array}. \quad (5.7)$$

5.1.1 Merging edges

Since G is a multigraph we may sometimes end up with multiple edges between the same set of vertices, i.e. we may have part of the TN on the form

$$\begin{array}{c} i_1 \quad k_1 \quad j_1 \\ \diagdown \quad \diagup \quad \diagdown \quad \diagup \\ \textcircled{T} \quad \textcircled{V} \\ \diagup \quad \diagdown \quad \diagup \quad \diagdown \\ i_m \quad k_r \quad j_n \end{array} = T^{i_1, \dots, i_m}_{k_1, \dots, k_r} V^{k_1, \dots, k_r}_{j_1, \dots, j_n}. \quad (5.8)$$

So long as the dimensionality of the modes are allowed to change, any TN on this form can always be reduced to an equivalent TN with only one edge joining the two tensors;

$$\begin{array}{ccc}
 \begin{array}{c} i_1 \\ i_2 \\ \vdots \\ i_m \end{array} & \begin{array}{c} k_1 \\ k_2 \\ \vdots \\ k_r \end{array} & \begin{array}{c} j_1 \\ j_2 \\ \vdots \\ j_n \end{array} \\
 \text{---} T \text{---} & \text{---} V \text{---} & \\
 \end{array} = \begin{array}{ccc}
 \begin{array}{c} i_1 \\ i_2 \\ \vdots \\ i_m \end{array} & K & \begin{array}{c} j_1 \\ j_2 \\ \vdots \\ j_n \end{array} \\
 \text{---} T' \text{---} & \text{---} V' \text{---} & \\
 \end{array}, \quad (5.9)
 \end{array}$$

where $T^{i_1, \dots, i_m}_K \equiv T^{i_1, \dots, i_m}_{k_1, \dots, k_r}$ and $K \equiv \varphi(k_1, \dots, k_r)$ with φ being any bijective map between $[d_1] \times [d_2] \times \dots \times [d_r]$ and $[\prod_l d_l]$. Hence, the dimension of the merged subspace is the product of the dimensions of the spaces being merged.

Evidently, the same procedure may be applied to the dangling edges from a vertex, so that any TN has an equivalent description as a simple open graph. The TN of eq (5.9) may for example be represented equally well as

$$I \text{---} T'' \text{---} K \text{---} V'' \text{---} J, \quad (5.10)$$

where indices I and J are mapped from $\{i_l\}_{l=1}^m$ and $\{j_l\}_{l=1}^n$ in a identical manner to that of K . In multilinear algebra, this operation of taking a multi-mode tensor T to a matrix T'' is known as a *matricization*, or *flattening* of T .

If we consider the adjoint of a matrix to be equivalent to the matrix itself, the number M of inequivalent matricizations of an n -mode tensor is equal to the number of ways to form two non-empty partitions out of a set of n elements, i.e. $M = 2^{n-1} - 1$.

We will be interested in the ranks of these matrices, which for a given n -mode tensor form an M -tuple of integers (r_1, r_2, \dots, r_M) . Failing to find a standard term, we will refer to this M -tuple as the *multi-rank* of a tensor, as it is a slight expansion on the similar multilinear rank.

5.1.2 Taking derivatives

Since a TN is linear in all its component tensors, taking partial derivatives with respect to the tensors becomes quite straightforward. Given a TN where the tensor T occurs only once, we may write its value as

$$\Psi = T \Psi_T, \quad (5.11)$$

where Ψ_T is the rest of the network, often referred to as the *environment* of T , and the multiplication is inner products taken over all modes of T . The partial derivative hence becomes simply

$$\frac{\partial \Psi}{\partial T} = \Psi_T. \quad (5.12)$$

As an illustrative example consider

$$\Psi = \begin{array}{c} \text{---} T \text{---} \\ \text{---} \cdot \text{---} \\ \text{---} \cdot \text{---} \end{array}, \quad (5.13)$$

where the smaller dots are arbitrary tensors. The derivative of Ψ with respect to T is then

$$\frac{\partial \Psi}{\partial T} = \Psi_T = \begin{array}{c} \text{---} \cdot \text{---} \\ \text{---} \cdot \text{---} \end{array}. \quad (5.14)$$

Note especially that the dangling edge of T becomes an identity operator in the corresponding mode of Ψ_T .

If T occurs $n > 1$ times in the TN, the situation becomes less trivial. In this case we may express Ψ as

$$\Psi = T^{\otimes n} \Psi_{T^{\otimes n}}. \quad (5.15)$$

Since the environments of the T 's at different sites are in general different, we introduce a labelling of these, denoting the T at site $i = 1, \dots, n$ with T_i . Hence,

$$\Psi = \left(\bigotimes_{i=1}^n T_i \right) \Psi_{T^{\otimes n}}. \quad (5.16)$$

Leibniz' rule then gives the sought derivative as

$$\frac{\partial \Psi}{\partial T} = \sum_{i=1}^n \left(\bigotimes_{j \neq i} T_j \right) \Psi_{T^{\otimes n}} = \sum_{i=1}^n \Psi_{T_i}, \quad (5.17)$$

where $\Psi_{T_i} \equiv \left(\bigotimes_{j \neq i} T_j \right) \Psi_{T^{\otimes n}}$ is the environment of T_i .

5.2 Entanglement in tensor networks

We will now take a look at how we may estimate the entanglement entropy of a TN state. To compute the exact Rényi entropies S_α with $\alpha > 0$, we would of course need to know the full set of tensors \mathcal{T} , in which case the calculation is rather straight forward and not that different from the generic case. What is interesting, however, is that it suffices to know the dimensionalities of the modes of each tensor $T_v \in \mathcal{T}$ to be able to give an upper bound on $S_0(A)$ and hence $S_\alpha(A)$ for any region $A \subseteq \partial E$.

We will also see that, for networks where all tensor mode dimensions are powers of the same integer, this limit is also realized for all configurations of \mathcal{T} except a set with Lebesgue measure zero. Both of these results goes under the name of Quantum max flow/min cut and are shown in [25], whose presentation we will summarise in this section.

5.2.1 Quantum max flow/min cut

Consider an open multigraph $G = (V, \varepsilon, E)$, and partition the edge points ε into two disjoint sets S, T , which we will refer to respectively as the sources and sinks of G . For any $u \in \varepsilon$, denote the edge connected to u as $e(u)$, and, for each edge $e \in E$, introduce a Hilbert space \mathcal{H}^e with dimension c_e . Anticipating that c_e will play a similar role to the capacity (or weights) in the classical max flow/min cut theorem, we introduce the *quantum capacity* as a function $c : E \rightarrow \mathbb{Z}_+$, such that $c(e) = c_e$.

From this setup, we may instantiate a TN by any assignment $v \mapsto T_v \in \mathcal{H}_v \equiv \bigotimes_{e \in \text{dv}} \mathcal{H}^e$. We denote the set of tensors from one such assignment by $\mathcal{T} \in \mathcal{I} \equiv \bigcup_{v \in V} \mathcal{H}_v$. Commonly, however, some or all the tensors will have additional constraints. In these cases we will instead write $\mathcal{T} \in \mathcal{J} \subseteq \mathcal{I}$.

In the context of our flow network, we may interpret the resulting TN from a particular choice of \mathcal{T} as a linear map $\beta(\mathcal{T})$ from source to sink: Let

$$\begin{cases} \mathcal{H}_S \equiv \bigotimes_{u \in S} \mathcal{H}^{e(u)}, \\ \mathcal{H}_T \equiv \bigotimes_{u \in T} \mathcal{H}^{e(u)}. \end{cases} \quad (5.18)$$

Then $\beta(\mathcal{T}) \in \text{hom}(\mathcal{H}_S, \mathcal{H}_T)$.

It will also be useful to more generally define, for any set of edges $A \subseteq E$ and any set of vertices $B \subseteq V$,

$$\begin{cases} \mathcal{H}_A \equiv \bigotimes_{e \in A} \mathcal{H}^e, \\ \mathcal{H}_B \equiv \bigotimes_{v \in B} \mathcal{H}_v. \end{cases} \quad (5.19)$$

Definition 5.1. The quantum max flow for a graph G with sources S , sinks T and quantum capacity c as above is

$$\text{QMF}(G, c; \mathcal{J}) \equiv \max_{\mathcal{T} \in \mathcal{J}} \text{rank } \beta(\mathcal{T}). \quad (5.20)$$

We will further take $\text{QMF}(G, c)$ to mean $\text{QMF}(G, c; \mathcal{I})$, from which it follows trivially that $\text{QMF}(G, c; \mathcal{J}) \leq \text{QMF}(G, c)$.

Given an edge cut set C disconnecting S from T , we further define the quantum capacity of C as

$$\text{QC}(C, c) \equiv \prod_{e \in C} c_e, \quad (5.21)$$

leading to the following definition:

Definition 5.2. The quantum min cut for a graph G with sources S , sinks T and quantum capacity c as above is

$$\text{QMC}(G, c) \equiv \min_{C \in \mathcal{C}} \text{QC}(C, c), \quad (5.22)$$

where \mathcal{C} is the set of edge cuts separating S from T .

As a quick aside, we may note that if we introduce a weight function $w(e) \equiv \log c(e)$, the weight of a cut C is equal to the logarithm of the quantum cut

$$|C| = \log \text{QC}(C), \quad (5.23)$$

also giving

$$\text{MC}(G) = \log \text{QMC}(G). \quad (5.24)$$

Lemma 5.1. Let C be an edge cut in G , separating S from T , and define $\mathcal{H}_C \equiv \bigotimes_{e \in C} \mathcal{H}^e$. Then

$$\beta(\mathcal{T}) = \beta_2 \beta_1, \quad (5.25)$$

where $\beta_1 \in \text{hom}(\mathcal{H}_S, \mathcal{H}_C)$ and $\beta_2 \in \text{hom}(\mathcal{H}_C, \mathcal{H}_T)$.

Proof. Form a partition (\bar{S}, \bar{T}) of $\bar{V} = V \cup \varepsilon$ such that $S \subseteq \bar{S}$ and $T \subseteq \bar{T}$, and C is set of edges connecting \bar{S} with \bar{T} . Replace each edge $(u, v) \in C$, where $u \in \bar{S}$ and $v \in \bar{T}$, with two half-edges (u, ε_u) and (v, ε'_u) . Denoting the sets of thus introduced edge points as U and U' , we arrive at two separated subgraphs G_1, G_2 such that \bar{S} is contained within G_1 and \bar{T} within G_2 .

Evidently, each G_i forms a TN in its own right, G_1 with sources S and sinks U and G_2 with sources U' and sinks T . Their linear maps become $\beta_1(\mathcal{T}_1) \in \text{hom}(\mathcal{H}_S, \mathcal{H}_C)$ and $\beta_2(\mathcal{T}_2) \in \text{hom}(\mathcal{H}_C, \mathcal{H}_T)$, where \mathcal{T}_i is the restriction of \mathcal{T} to G_i and we have used that $\mathcal{H}_C = \mathcal{H}_U = \mathcal{H}_{U'}$.

The map $\beta(\mathcal{T})$ of G is then simply the composition of $\beta_1(\mathcal{T}_1)$ and $\beta_2(\mathcal{T}_2)$, i.e. $\beta(\mathcal{T}) = \beta_2 \beta_1$, proving the lemma. \square

Corollary 5.1. $\text{QMF}(G, c) \leq \text{QMC}(G, c)$

Proof. This follows immediately from the previous lemma. We can write $\beta = \beta_2 \beta_1$ with β_i defined as above, for any \mathcal{T} and cut C . Choose the decomposition corresponding to the minimal cut C_{\min} w.r.t. c . Then

$$\text{QMF}(G, c) = \max_{\mathcal{T}} \text{rank } \beta = \max_{\mathcal{T}} \text{rank}(\beta_2 \beta_1) \leq \max_{\mathcal{T}_1} \text{rank } \beta_1 \leq \dim \mathcal{H}_{C_{\min}} = \text{QMC}(G, c).$$

□

While the above result is well known, the lower bound on QMF is less studied. The following theorem shows that $\text{QMC} = \text{QMF}$ for special capacity functions.

Theorem 5.1. Quantum max-flow/min-cut theorem

If there is a positive integer $d \in \mathbb{Z}_+$ such that $\log_d(c_e) \in \mathbb{Z}_+$ for all $e \in E$, then

$$\text{QMC}(G, c) = \text{QMF}(G, c) \tag{5.26}$$

Proof. This can be proven by constructing an explicit instantiation of \mathcal{T} which fulfils $\text{rank } \beta(\mathcal{T}) = \text{QMC}(G, c)$, since the previous corollary shows $\text{QMF} \leq \text{QMC}$. For details, see [25]. □

This actually gives a lower bound on QMF for any TN, since we can choose \mathcal{T} in such a way that each $c(e)$ is effectively reduced to be d taken to some integer power, for some $d \in \mathbb{Z}_+$.

Corollary 5.2. If there is a positive integer $d \in \mathbb{Z}_+$ such that $\log_d(c_e) > m_e \in \mathbb{Z}_+$ for all $e \in E$, then

$$\text{QMF}(G, c) \geq \text{QMC}(G, c') \tag{5.27}$$

where $c'(e) \equiv d^{m_e}$.

Proof. Taking $c \rightarrow c'$ demands us to make a corresponding transformation $\mathcal{H}^e \rightarrow \mathcal{H}'^e$ of the Hilbert spaces, since $c(e) \equiv \dim \mathcal{H}^e$ and $c'(e) \equiv \dim \mathcal{H}'^e$. Choose $\mathcal{H}'^e \subseteq \mathcal{H}^e$, which is possible due to that $c'(e) \leq c(e) \forall e$.

Each tensor of the original TN is then an element $T_v \in \mathcal{H}_v = \bigotimes_{e \in \text{dv}} \mathcal{H}^e$, while the tensors of the reduced TN are $T'_v \in \mathcal{H}'_v = \bigotimes_{e \in \text{dv}} \mathcal{H}'^e$. Hence, $\mathcal{H}'_v \subseteq \mathcal{H}_v \forall v \in V$, from which it follows directly that $\text{QMF}(G, c) \geq \text{QMF}(G, c')$, and by theorem 5.1, $\text{QMF}(G, c') = \text{QMC}(G, c')$, which finishes the proof. □

It is also possible to show that $\text{QMF} = \text{QMC}$ for a slightly different special case of TN's:

Proposition 5.1. Let G, c be as above and introduce a weight function $w(e) \equiv \log_d c(e)$ for some $d > 1$. Consider the classical max flow $\text{MF}(G, w)$. If this flow is at each edge \log_d of some integer and there is no closed loop in G in which all edges have non-zero flow, then $\text{QMF}(G, c) = \text{QMC}(G, c)$.

Proof. See [25] □

While the above results are interesting in their own right, their usefulness is quite limited, due to the fact that QMF is stated as a maximisation over \mathcal{T} , but the more common case is to have a fixed \mathcal{T} . However, the following proposition asserts that the maximum quantum flow is actually reached for almost all choices of \mathcal{T} .

Proposition 5.2. The set \mathcal{K} of all tensors $\mathcal{T} \in \mathcal{I}$ such that $\text{rank } \beta(\mathcal{T}) = \text{QMF}(G, c)$ is an open, dense subset of \mathcal{I} .

Proof. Let $M \equiv \text{QMF}(G, c)$. We note that $\text{rank } \beta(\mathcal{T}) < M$ implies the vanishing of all $M \times M$ minors to β . The minors are polynomials, meaning that $\mathcal{K}^c = \{\mathcal{T}; \text{rank } \beta(\mathcal{T}) < M\}$ is a proper affine variety. This tells us [25, 44] that \mathcal{K} is an open, dense subset of \mathcal{I} , which is the statement of the proposition. \square

5.2.2 Quantum max flow/min cut with restricted tensors

It is a common case that one wishes to consider tensor networks where the tensors are chosen from a smaller set than \mathcal{I} . For this reason, we make some minor extensions to the ideas of Cui et al [25] in this section, beginning with a slight generalization of the preceding proposition.

Fix a basis $\{e_i^e\}_{i=1}^{c_e}$ for each edge space \mathcal{H}^e with $e \in E$. This implies a basis $\{e_i^v\}$ for each vertex space \mathcal{H}_v , $v \in V$, in which we may expand each tensor T_v according to

$$T_v = \sum_i T_v^i e_i^v, \quad T_v^i \in \mathbb{F} \quad (5.28)$$

where \mathbb{F} is the field of the Hilbert space. Introduce an arbitrary ordering of these components according to $t_{\varphi(v, i)} \equiv T_v^i$, where $\varphi(v, i)$ is an arbitrary bijection mapping vertex-index pairs onto $[N]$ with N being the number of such pairs. We then have $t \in \mathbb{F}^N$.

Fix a subset of the components T_v^i , that is, partition t into $t = (t_{\text{free}}, t_{\text{fixed}})$ such that $t_{\text{free}} \in \mathbb{F}^n$ and $t_{\text{fixed}} = \alpha$ for some constant $\alpha \in \mathbb{F}^{N-n}$.

Now, let $\mathcal{J} \subseteq \mathcal{I}$ be the space in which \mathcal{T} now takes values, i.e. let

$$\mathcal{J} \equiv \{\mathcal{T} \in \mathcal{I}; t_{\text{free}} \in \mathbb{F}^n, t_{\text{fixed}} = \alpha\} \cong \mathbb{F}^n. \quad (5.29)$$

Proposition 5.3. The set \mathcal{K} of all tensors $\mathcal{T} \in \mathcal{J}$ such that $\text{rank } \beta(\mathcal{T}) = \text{QMF}(G, c; \mathcal{J})$ is an open, dense subset of \mathcal{J} .

Proof. The proof is completely analogous to that of prop. 5.2, since all minors of β are polynomials in the components of t_{free} . \square

A commonly occurring restriction is that one or more of the tensors are diagonal. If these are adjacent to minimal cuts it will reduce the quantum max flow of the TN.

Proposition 5.4. Let \mathcal{J} be a subset $\mathcal{J} \subseteq \mathcal{I}$ such that T_v is restricted to be a diagonal tensor for some vertex $v \in V$. For simplicity we demand in addition that all bond dimensions $c_e = \dim \mathcal{H}_e$ for $e \in dv$ are equal to d . Consider the cut set C of an arbitrary quantum minimal cut $\text{QMC}(G, c)$ and define $m \equiv \max\{|C \cap dv| - 1, 0\}$. Then

$$\text{QMF}(G, c; \mathcal{J}) \leq d^{-m} \text{QMC}(G, c). \quad (5.30)$$

Proof. We begin by noting that for $m = 0$, the statement is identical to the limit we have already shown for $\text{QMF}(G, c)$ with unrestricted tensors.

Consider instead the case $m \geq 1$, and once again note that the cut set defines an intermittent Hilbert space $\mathcal{H}_C \equiv \bigotimes_{e \in C} \mathcal{H}_e$, so that the map $\beta(\mathcal{T})$ hence may be partitioned into $\beta = \beta_T \beta_S$, with $\beta_S \in \text{hom}(\mathcal{H}_S, \mathcal{H}_C)$ and $\beta_T \in \text{hom}(\mathcal{H}_C, \mathcal{H}_T)$.

We may without loss of generality assume T_v to be a part of β_T , which we may then partition according to $\beta_T = \beta_{T'} \beta_v$, with $\beta_v = T_v \otimes \text{id}(\mathcal{H}_{C \setminus dv}) \in \text{hom}(\mathcal{H}_C, \mathcal{H}_{C \setminus dv \cup dv \setminus C})$. This gives $\beta(\mathcal{T}) = \beta_{T'} \beta_v \beta_S$, and hence

$$\text{rank } \beta(\mathcal{T}) \leq \text{rank } \beta_v = \dim \mathcal{H}_{C \setminus dv} \text{rank } T_v = d^{-m-1} \text{QMC}(G, c) \text{rank } T_v, \quad (5.31)$$

where we by $\text{rank } T_v$ mean the matrix rank of the flattening of T_v implied by β_v . However, it is a simple result from multilinear algebra that the multi-rank of a diagonal d^n -dimensional tensor is (r, r, \dots, r) , where $r \leq d$ is the number of non-zero elements of the diagonal vector. We hence have $\text{rank } T_v \leq d$, giving

$$\text{rank } \beta(\mathcal{T}) \leq d^{-m} \text{QMC}(G, c). \quad (5.32)$$

This holds for any assignment of tensors $\mathcal{T} \in \mathcal{J}$, which is precisely what is required by the statement of the proposition. \square

Proposition 5.5. Let \mathcal{J} be a subset $\mathcal{J} \subseteq \mathcal{I}$ such that T_{v_i} for $i = 1, \dots, n$ are restricted to be diagonal tensors for some non-adjacent vertices $v_i \in V$. We further demand that each tensor T_{v_i} have all bond dimensions c_e for $e \in dv_i$ equal to d_i .

Consider the cut set C of an arbitrary quantum minimal cut $\text{QMC}(G, c)$ and define $m_i \equiv \max\{|C \cap dv_i| - 1, 0\}$. Then

$$\text{QMF}(G, c; \mathcal{J}) \leq \prod_{i=1}^n d_i^{-m_i} \text{QMC}(G, c). \quad (5.33)$$

Proof. The proof is analogous to the preceding proposition, but with β_v replaced by $\beta_{\{v_i\}} = (\otimes_i T_{v_i}) \otimes \text{id}(\mathcal{H}_{C \setminus \cup_i dv_i})$. \square

One thing to be noted about the latter proposition is that the demand of the vertices to be non-adjacent is essential to ensure that β may indeed be written on the form $\beta_{T'} \beta_{\{v_i\}} \beta_S$, with $\beta_{\{v_i\}}$ as indicated. The non-adjacency demand should however never be any problem in a practical situation, since the contraction of two adjacent diagonal tensors is itself a diagonal tensor.

The above two propositions points us toward a modified notion of quantum capacity for the cases where some tensors are restricted to be diagonal; especially the cumbersomeness of having to restrict to non-adjacent diagonal tensors give us a clue that the more natural notion of a cut is for this problem rather the combined cut discussed in section 4.4. We will now proceed to introduce such a notion for TNs with diagonal tensors.

Consider now a TN on the network $G = (V, E)$, which is restricted such that all the tensors on the vertices $U \subset V$ are diagonal and denote the restriction of \mathcal{I} that this implies with \mathcal{J} . Further let dv be the set of edges connected to any vertex in U and with ∂C the set of vertices connected to any edge in the set of edges $C \subseteq E$.

The capacity function is now taken to be $c : V \cup E \rightarrow \mathbb{Z}_+ \cup \infty$, with $c(e) \equiv \dim \mathcal{H}^e$ for all $e \in E$, as usual. For vertices $v \in V$ on the other hand, we define

$$c(v) \equiv \begin{cases} \text{rank } T_v & \text{if } v \in U, \\ \infty & \text{otherwise.} \end{cases} \quad (5.34)$$

Note that, for all $v \in U$, $c(v) = \text{rank } T_v \leq \min_{e \in dv} c(e)$, by virtue of T_v being diagonal.

We may now proceed to introduce our modified definitions of the quantum cuts.

Definition 5.3. Given a combined cut set $C = C_V \cup C_E$, with $C_V \subset V$ and $C_E \subset E$, disconnecting S from T , we define the restricted quantum capacity

$$\text{QC}(C, c; U) \equiv \prod_{x \in C} c(x). \quad (5.35)$$

We further define the restricted quantum minimal cut to be

$$\text{QMC}(G, c; U) \equiv \min_{C \in \mathcal{C}} \text{QC}(C, c; U), \quad (5.36)$$

where \mathcal{C} is the set of all combined cut sets separating S from T .

This allows us to introduce this slightly more general version of prop. 5.5:

Proposition 5.6. $\text{QMF}(G, c; \mathcal{J}) \leq \text{QMC}(G, c; U)$.

Proof. Similar to that of prop. 5.5. □

In fact, it turns out that with the above notion of quantum min cut, the quantum max flow/min cut theorem holds also for our restricted TN case.

Theorem 5.2. (Restricted quantum max flow/min cut theorem)

If no two vertices $u, v \in U$ are adjacent and there is a positive integer d such that $\log_d c(x) \in \mathbb{Z}_+$ for all $x \in E \cup U$, then

$$\text{QMF}(G, c; \mathcal{J}) = \text{QMC}(G, c; U). \quad (5.37)$$

Proof. Since we already know $\text{QMF}(G, c; \mathcal{J}) \leq \text{QMC}(G, c; U)$, it is sufficient to construct an explicit map $\beta(\mathcal{T})$ such that $\text{rank } \beta(\mathcal{T}) = \text{QMC}(G, c; U)$ and $\mathcal{T} \in \mathcal{J}$. To keep the construction relatively sane we will restrict to the case of all diagonal tensors being Kronecker deltas; that is, denoting the rank n Kronecker delta by

$$\delta_n \equiv \text{Diag}\{\underbrace{1, 1, \dots, 1}_{n \text{ elements}}\}, \quad (5.38)$$

leaving the number of modes implicit, we set $T_v \equiv \delta_{c(v)} \forall v \in U$. The generalization to arbitrary diagonal tensors is conceptually no different.

We first note that we may without loss of generality truncate the Hilbert spaces of each diagonal tensor to have the same dimension as the rank of the tensor; i.e. it suffices to consider the case where for each $v \in U$, $e \in dv$ we have $c(e) = c(v) \equiv \text{rank } T_v$.

The next step is then to construct an equivalent TN on a graph $G' = (V', E')$ where all edges have equal quantum capacity. If edge $e = (u, v)$ has $c_e = d^m$ this is achieved by replacing e with m parallel edges $\{e_1, \dots, e_m\}$ between u and v each with quantum capacity d . Note that in particular, this turns the Kronecker delta tensor δ_{d^m} into $(\delta_d)^{\otimes m}$, so that a vertex $u \in U$ with tensor δ_{d^m} maps into m vertices $\{u_i\}_{i=1}^m$ in U' , each fulfilling $du_i = du$ and having the tensor δ_d .

Denoting the set of vertices with diagonal tensors in G' with U' , we thus arrive at the following modified capacity function;

$$c'(x) \equiv \begin{cases} d & \text{if } x \in E' \cup U' \\ \infty & \text{otherwise.} \end{cases} \quad (5.39)$$

We will also for the sake of clarity introduce the unweighted capacity function

$$c''(x) \equiv \begin{cases} 1 & \text{if } x \in E' \cup U' \\ \infty & \text{otherwise.} \end{cases} \quad (5.40)$$

Since by construction $\text{QMC}(G', c'; U') = \text{QMC}(G, c; U)$, the *unweighted* min cut of the equal-weight TN becomes $\text{MC}(G', c'') = \log_d \text{QMC}(G, c; U)$. By theorem 4.6, we thus have exactly $p \equiv \text{MC}(G', c'')$ edge-disjoint paths from S to T such that the maximum number of paths through $v \in V'$ is $c''(v)$. Make a choice of p such paths P and assign the tensors T'_v with $v \in V' \setminus U$ in the following manner:

Let P_v be the set of length 2 path segments (e_1, e_2) in the paths P such that $e_1, e_2 \in dv$, and define for $e \in dv$

$$\mathbf{x}_e \equiv \begin{cases} \mathbf{x}_0 \equiv (1, 0, \dots, 0) & \text{if } e \cap U = \emptyset \\ \mathbf{x}_1 \equiv (1, 1, \dots, 1) & \text{otherwise.} \end{cases} \quad (5.41)$$

Then T'_v is assigned the value

$$T'_v \equiv \left(\bigotimes_{(e_1, e_2) \in P_v} \delta_d(\mathcal{H}^{e_1}, \mathcal{H}^{e_2}) \right) \left(\bigotimes_{e \in dv \setminus P} \mathbf{x}_e \right). \quad (5.42)$$

This construction can then be seen to put β on the form

$$\beta = \alpha (\delta_d)^{\otimes p} (\mathbf{x}_0)^{\otimes n} \neq 0, \quad (5.43)$$

where $\alpha \in \mathbb{Z}_+$ and n is such that the dimensionality of β agrees with the capacity function. From this it is trivial to read out $\text{rank } \beta = d^p = \text{QMC}(G, c; U)$. Since any β which is realizable on G' with diagonal tensors on U' , is also realizable on G with diagonal tensors on U , this proves the theorem. \square

Once we have shown this, it is a trivial matter to show that the analogous statement to corollary 5.2 also holds.

Corollary 5.3. If there is a positive integer $d \in \mathbb{Z}_+$ such that $\log_d(c(x)) \geq m_x \in \mathbb{Z}_+$ for all $x \in E \cup U$, then

$$\text{QMF}(G, c; \mathcal{J}) \geq \text{QMC}(G, c'; U) \quad (5.44)$$

where $c'(x) \equiv d^{m_x}$.

Proof. Identical to that of corollary 5.2. \square

5.2.3 Quantum max flow and entanglement entropy

A large part of our interest in studying quantum max flow/min cut comes from the fact that we can use it as a simple short cut to estimating the quantum entropy of a given TN state.

First consider a mixed state described by the density matrix $\rho \in \text{end}(\mathcal{H})$, for some arbitrary Hilbert space \mathcal{H} . Since ρ is a linear map, it is always expressible as a TN on some graph G , with capacity function c and tensors \mathcal{T} . It is easy to see that in this description, $\beta(\mathcal{T}) = \rho$, from which it follows that the zeroth Rényi entropy is

$$S_0(\rho) = \log \text{rank } \rho = \log \text{rank } \beta(\mathcal{T}) \leq \log \text{QMF}(G, c), \quad (5.45)$$

and, since S_0 is an upper bound on S_α for $\alpha \geq 0$, this gives

$$S_\alpha(\rho) \leq \log \text{QMF}(G, c). \quad (5.46)$$

We might instead consider describing the pure state $|\psi\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$ by a tensor network G, c, \mathcal{T} . Letting A correspond to the sources and B to the sinks, this TN describes a map $\beta(\mathcal{T}) \in \text{hom}(\mathcal{H}_A, \mathcal{H}_B)$. The density matrix is $\rho \equiv |\psi\rangle\langle\psi|$ and the reduced density matrix for system A may then be expressed as

$$\rho_A = \text{tr}_B \rho = \beta^\dagger \beta, \quad (5.47)$$

introduced, perhaps the most common of which is the left/right *canonical* or *isometric* form. In left canonical form, with the right canonical defined similarly, each tensor $A_i^j_k$ is restricted to

$$A^{\dagger i}_j{}^k A_i^j{}_l = \text{id}^k{}_l, \quad \text{or simply} \quad \begin{array}{c} \boxed{A} \\ | \\ \boxed{A^\dagger} \end{array} = \left[\begin{array}{c} \\ \end{array} \right]. \quad (5.54)$$

This is easily seen to reduce the gauge freedom to $M^{(j)} \in U(d_j)$ where d_j is the dimension of the left virtual bond of $A^{(j)}$.

5.3.2 Entanglement

For locally interacting spin systems in two dimensions or more, the MPS construction with low bond dimensions generally stops being efficient. This is easily seen to be the case by considering the entropy $S(A)$ of some connected region A of the system. For physical states this tends to scale as the area of the boundary of A , i.e. in D dimensions, $S(A) \propto L^{D-1}$, where L is the length scale of A . On the other hand, for an MPS with bond dimensions $d_i \leq d$ for all i , the quantum min cut/max flow theorem immediately gives $S(A) \leq 2 \log d$, suggesting that the MPS description can only be accurate for $d \propto e^{L^{D-1}}$, which is problematic for all $D > 1$.

6

Numerical renormalization group methods in quantum mechanics

The Renormalization group (RG) is a set of methods from theoretical physics dealing with transformations between different length and/or energy scales. These transformations are typically defined such that the theories become self-similar at different scales, but parameters of the theory are allowed to vary. This self-similarity often implies that information is lost as the length scale is increased, rendering the transformations non-invertible. Of particular interest is the fixed points of the RG transformations, where the theory becomes scale invariant.

While RG ideas are omnipresent throughout theoretical physics, we will in this thesis mainly concern ourselves with the subset dealing with numerical simulations of quantum many-body systems on lattices, most notably the Density Matrix Renormalization group by White [14, 15] and the Entanglement Renormalization procedure by Vidal [16], which is perhaps best recognized by the name of the related quantum state ansatz, MERA.

6.1 Numerical real-space renormalization group

A simple view of real-space RG which we will now present is done by Vidal [16], although the basic ideas date back to Wilson [46] and White [14]. Consider a quantum mechanical system on a D -dimensional lattice \mathcal{L} , which is described by a state $|\psi\rangle$ in a Hilbert space

$$\mathcal{H}_{\mathcal{L}} \equiv \bigotimes_{s \in \mathcal{L}} \mathcal{H}_s, \quad (6.1)$$

where \mathcal{H}_s is the finite-dimensional Hilbert space of site $s \in \mathcal{L}$. It will also be useful to define

$$\mathcal{H}_A \equiv \bigotimes_{s \in A} \mathcal{H}_s \quad (6.2)$$

for any set of sites $A \subset \mathcal{L}$.

An elementary real-space RG transformation is then a transformation from the lattice \mathcal{L} to a smaller *effective* lattice \mathcal{L}' , in which each site $s' \in \mathcal{L}'$ corresponds to a *coarse-grained* version of a block $B \subset \mathcal{L}$ of neighbouring sites in \mathcal{L} . This means that for a site $s' \in \mathcal{L}'$, the corresponding Hilbert space is chosen as

$$\mathcal{H}'_{s'} \equiv \mathcal{S}_B \subseteq \mathcal{H}_B, \quad (6.3)$$

where the map between $\mathcal{H}'_{s'}$ and \mathcal{H}_B as well as the subspace \mathcal{S}_B is induced by means of an isometry

$$w : \mathcal{H}'_{s'} \rightarrow \mathcal{H}_B. \quad (6.4)$$

Note that this isometry implies $w^\dagger w = \text{id}$ but that $ww^\dagger \neq \text{id}$ unless w is also unitary, which would imply $\mathcal{S}_B = \mathcal{H}_B$.

Since an isometry between Hilbert spaces is a linear map, we may express w as a tensor, which for three-site blocks, $B = \{s_1, s_2, s_3\}$, may be expressed in TN notation as

$$w = \begin{array}{c} s' \\ \triangle \\ s_1 \quad s_2 \quad s_3 \end{array}, \quad \text{where} \quad w^\dagger w = \begin{array}{c} \triangle \\ w \\ \triangle \\ w^\dagger \\ \triangle \end{array} = \text{id}, \quad (6.5)$$

with obvious generalizations to different block sizes.

The question that remains is now how to choose w and in particular \mathcal{S}_B in such a way as to minimize the dimension m of $\mathcal{H}'_{s'}$, while keeping the relevant properties of $|\psi\rangle$ in the renormalized state $|\psi'\rangle \in \mathcal{H}'_{\mathcal{L}'}$.

As observed by [14], this is most clearly phrased in terms of the reduced density matrix of the block B ,

$$\rho_B \equiv \text{tr}_{B^c} |\psi\rangle \langle \psi|, \quad (6.6)$$

in terms of which the renormalization step becomes

$$\rho_B \rightarrow \rho_{s'} = w^\dagger \rho_B w = \begin{array}{c} \triangle \\ w \\ \rho_B \\ w^\dagger \\ \triangle \end{array}. \quad (6.7)$$

We may then phrase the optimal choice of w as the one that minimizes m while projecting out as little as possible of the probability density of ρ_B . In other words, for a tolerance of ϵ , we choose the w which minimizes m while fulfilling

$$\text{tr} \rho_B - \text{tr} \rho_{s'} \leq \epsilon \quad (6.8)$$

This condition immediately leads to

$$\mathcal{S}_B = \text{span}\{|p_1\rangle, |p_2\rangle, \dots, |p_m\rangle\}, \quad (6.9)$$

where $|p_i\rangle$ is the eigenvector corresponding to the i th largest eigenvalue p_i of ρ_B and m is chosen to be the smallest integer for which (6.8) holds.

6.2 Density matrix renormalization group

The density matrix renormalization group (DMRG) is an algorithm originally introduced by White [14, 15] in the context of real-space RG, but in modern literature more often interpreted as a variational approach for numerical optimization of many-particle quantum states. It is arguably one of the most successful numerical methods within the field of

condensed matter, and have been applied with good results for a large range of problems, especially in the context of strongly correlated systems on 1D lattices. We will make a brief introduction of the method, following [47].

As in the previous section, we consider a system on a 1-dimensional lattice \mathcal{L} , with quantum mechanical state space $\mathcal{H}_{\mathcal{L}} \equiv \bigotimes_{s \in \mathcal{L}} \mathcal{H}_s$. For concreteness we specialize to the problem of finding the ground state of some local Hamiltonian H .

6.2.1 Infinite lattice DMRG

The infinite lattice DMRG starts by considering the system on a lattice of length small enough that the ground state may be calculated exactly. Partition this smaller lattice into a left block A and a right block B where A and B is of equal size, such that $\mathcal{L} = A B$, where we by $A B$ simply mean the concatenation of A with B . Now enlarge the lattice with two sites between the blocks, to

$$\mathcal{L}' \equiv A \bullet \bullet B. \quad (6.10)$$

In this slightly larger lattice we may write an arbitrary state as

$$|\psi\rangle = \sum_{a,s_A,s_B,b} \psi_{a,s_A,s_B,b} |a\rangle_A |s\rangle_A |s\rangle_B |b\rangle_B \quad (6.11)$$

where $|a\rangle_A$ forms a basis of A , $|b\rangle_B$ forms one over B , and $|s\rangle_A$ and $|s\rangle_B$ are bases for the left and right newly introduced site respectively.

Given that we chose our initial lattice sufficiently small, we may now find the ground state of our Hamiltonian on \mathcal{L}' , by finding the $|\psi\rangle$ that minimizes

$$E = \frac{\langle \psi | H | \psi \rangle}{\langle \psi | \psi \rangle}. \quad (6.12)$$

Having done this, we now wish to increase the size of the lattice. To allow for that, however, we need to make sure the Hilbert space dimension of the problem stays contractible. DMRG deals with this by forming new blocks $A\bullet$ and $\bullet B$ which are coarse-grained to A' and B' precisely as in section 6.1, where the dimension of the thus coarse-grained Hilbert space is selected either by some limit ϵ on the truncation error, or simply truncated to some maximum m .

The thus found isometries may then be applied also to the Hamiltonian, giving a coarse-grained approximation $H' = w_A^\dagger w_B^\dagger H w_A w_B$. This completes one step of the algorithm, which now can be rerun, introducing two new sites between A' and B' , with corresponding terms in the Hamiltonian. This process is illustrated in figure 6.1a and can of course be repeated indefinitely, until a desired lattice size L is reached.

6.2.2 Finite lattice DMRG

While there are some cases where the solution that is arrived at by the infinite lattice DMRG is sufficient, it does make the quite bold assumption that the ground states of smaller lattices will be more or less identical to those of larger lattices. To alleviate this assumption, one can run a few iterations of a variant on the above algorithm, known as *finite lattice DMRG*, where the lattice size is kept constant.

Just as before, the lattice is partitioned into $A\bullet\bullet B$, the ground state $|\psi\rangle$ of the approximated Hamiltonian is found, and $A\bullet$ is coarse-grained to A' . The other block, however, then has to shrink by one site in order to keep the lattice size for the next iteration. This

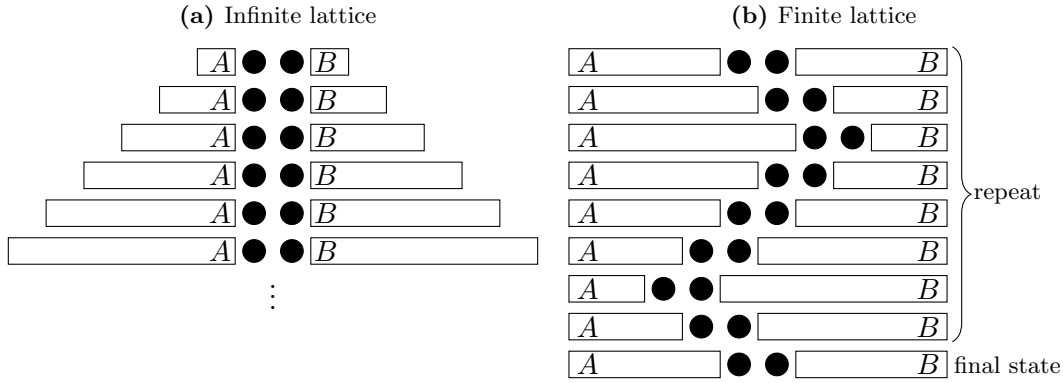


Figure 6.1: Illustration of the blocking procedure of the lattice in (a) infinite and (b) finite DMRG. Each line is to be interpreted as a step in the algorithm, which runs from top to bottom.

is done by taking B' to be the most recently calculated right-block of the desired size. The algorithm can now be repeated, until B is reduced to having zero sites, after which one switches to instead having B grow at the expense of A . Running the algorithm from small A to large A and then back to small is known as a *sweep*, and one typically only have to complete a few of them to get convergence. This procedure is illustrated in figure 6.1b.

6.2.3 DMRG and MPS

As originally found by [48, 49], both the infinite and finite DMRG may be expressed as variational algorithms over MPS states. To see this, denote the effective basis for block A with length l as $|a_l\rangle$, and for the site next to it by $|s_l\rangle$. Let their respective Hilbert spaces be \mathcal{H}_{A_l} and \mathcal{H}_{s_l} . As noted before, when growing a block from length $l-1$ to l , DMRG finds an isometry $w^{[l]}$ that is then used to map $\mathcal{H}_{A_{l-1}} \otimes \mathcal{H}_{s_l}$ into \mathcal{H}_{A_l} as

$$|a_l\rangle = \sum_{a_{l-1}, s_l} w^{[l]a_l}_{a_{l-1}, s_l} |a_{l-1}\rangle |s_l\rangle. \quad (6.13)$$

Expanding this recursively leads immediately to the (partial) MPS

$$|a_l\rangle = \sum_{\{s_i\}} w^{[1]} w^{[2]} \dots w^{[l]a_l} |s_1\rangle |s_2\rangle \dots |s_l\rangle = \begin{array}{c} \bullet \quad \bullet \quad \dots \quad \bullet \\ | \quad | \quad \dots \quad | \\ s_1 \quad s_2 \quad \dots \quad s_l \end{array} \quad a_l, \quad (6.14)$$

and the isometry conditions on $w^{[i]}$ immediately gives that this MPS is on left canonical form. Performing the same process for block B gives a corresponding MPS on right canonical form, so that the full state on $A \bullet \bullet B$ may be described as

$$|\Psi\rangle = \begin{array}{c} \bullet \quad \bullet \quad \dots \quad \bullet \\ | \quad | \quad \dots \quad | \\ \bullet \quad \bullet \quad \dots \quad \bullet \end{array}, \quad (6.15)$$

where

$$|\psi\rangle = \begin{array}{c} \bullet \\ | \\ \bullet \end{array} \quad (6.16)$$

is the state of the reduced Hilbert space, i.e. the same as that in equation (6.11). The optimization is then at each stage performed over $|\psi\rangle$. For the example problem of finding

have scale invariant systems as fixed points. Instead, running DMRG on these systems typically causes the state space to increase with each iteration until eventually running out of memory. This was noted by Vidal, causing him to propose the *Entanglement Renormalization* RG procedure[16].

A significant point that one may notice about the procedure of section 6.1 is that the minimal choice of m is highly dependent on the amount of entanglement between B and B^c . A simple way to see this is to note that for $\epsilon = 0$, $m = \text{rank } \rho_B$, which is simply the Schmidt number of ρ_B . For non-zero ϵ , this is of course not an exact relation, but the connection with entanglement entropy remains.

Based on this observation, Entanglement Renormalization is a method of reducing the dimensionality of the renormalized Hilbert space while preserving the locality of operators by performing unitary operations between boundary sites in B and adjacent sites in B^c so as to minimize short-range entanglement prior to coarse graining.

To formalize the procedure it is simplest to restrict ourselves to the case of three-site blocks $B = \{s_1, s_2, s_3\}$ on a 1-D lattice, noting that the generalization to larger block sizes will be conceptually no different. Let $r_1, r_3 \in B^c$ be the nearest neighbours of s_1 and s_3 respectively and define $A \equiv B \cup \{r_1, r_2\}$. We can then introduce two *disentangler* u_1 and u_3 as unitary operators on $\mathcal{H}_{r_1} \otimes \mathcal{H}_{s_1}$ and $\mathcal{H}_{r_3} \otimes \mathcal{H}_{s_3}$ respectively, chosen such that the entanglement of

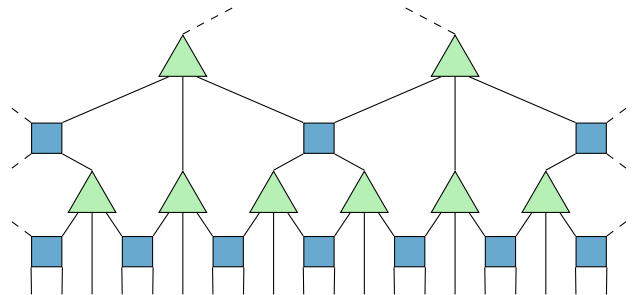
$$\tilde{\rho}_B \equiv \text{tr}_{r_1, r_3} [(u_1 \otimes u_3)^\dagger \rho_A (u_1 \otimes u_3)] \quad (6.22)$$

is smaller than or equal to that of ρ_B . Coarse graining with an isometry w as in the previous section then gives a truncated Hilbert space $\tilde{\mathcal{H}}_{\tilde{s}}$ with dimension $\tilde{m} \leq m$, which is of course desirable. We can express the full RG transformation of B as



$$\quad (6.23)$$

and performing the procedure recursively on the entire lattice leads to a transform on the form



$$\quad (6.24)$$

where the individual tensors need not be identical.

This form of TN (and generalizations to higher dimensions) can also be utilized as an ansatz for the quantum many-body state itself, and then goes under the name of *Multiscale Entanglement Renormalization Ansatz* (MERA) [50].

Part II

Machine learning

7

Machine learning

Machine learning may in general terms be defined as the study of algorithms which learn from data. This definition is of course quite vague, meaning that, to make any progress at all we will need to introduce several assumptions on the nature of the data and of the task, as well as a more stringent definition of what is meant by “learning”. One formalisation of this, which we will use, is known as *Statistical learning theory*. A more thorough presentation of the subject can be found in e.g. [51–53].

We will in this chapter introduce a multitude of different machine learning models, focusing mainly on representational aspects, while largely ignoring the equally (if not more) important issue of training. A good place to start if one desires a more covering review of (relatively) current machine learning, and deep learning in particular is the book by Goodfellow et al [37], or the somewhat more brief review article by Mehta et al [54].

7.1 Statistical learning theory

We may consider the problem of learning as the task of finding some optimal function $f : \mathcal{X} \rightarrow \mathcal{Y}$ based on empirical data \mathcal{D} . The starting point of statistical learning theory is to let this data consist of samples (z_1, z_2, \dots, z_N) with $z_i \in \mathcal{Z} \supseteq \mathcal{X}$, and to assume that the z_i are i.i.d. according to some distribution $P(z) \in \mathcal{P}(\mathcal{Z})$, where $\mathcal{P}(\mathcal{Z})$ is the set of probability distributions over \mathcal{Z} . Typically, the optimal f is considered to be that which minimizes some *risk functional* $R[f]$, often defined as some expected value of the error made by the function for particular samples.

Machine learning problems where the algorithm is fed pre-existing datasets may be broadly categorized into two classes, *supervised* and *unsupervised* learning. There is another class of problems grouped under the term *reinforcement learning*, where there is no pre-existing dataset and the algorithm thus has to learn by taking individual samples. We will only consider supervised and unsupervised learning, however.

7.1.1 Supervised learning

In supervised learning, the dataset is labelled, meaning that $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, and the function to learn is often taken as

$$f : \mathcal{X} \rightarrow \mathcal{Y}. \quad (7.1)$$

The $x_i \in \mathcal{X}$ are generally referred to as patterns or inputs, while $y_i \in \mathcal{Y}$ are labels or outputs. The risk functional is then generally expressed as

$$R[f] \equiv \int d(f(x), y) dP(x, y), \quad (7.2)$$

where $d : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+$ is some (quasi-)distance measure on \mathcal{Y} . Note that this implies that we may by appropriately choosing $d(\cdot, \cdot)$ arrive at an interpretation of the ideal $f(x)$ as the mode or mean of $P(y|x)$, depending on whether \mathcal{Y} is discrete.

Alternatively, we may want to find something like an estimate of the conditional distribution $P(y|x)$, in which case we let $f : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$ and set the risk functional to e.g.

$$R[f] \equiv \int d(f(x), P(\cdot|x)) dP(x), \quad (7.3)$$

where $P(x) \in \mathcal{P}(\mathcal{X})$ is the marginal distribution over x and $d : \mathcal{P}(\mathcal{Y}) \times \mathcal{P}(\mathcal{Y}) \rightarrow \mathbb{R}_+$.

Common examples of supervised learning include image classification, speech recognition and translation.

7.1.2 Unsupervised learning

In unsupervised learning, there is no division of the dataset into inputs and labels. Rather the input space \mathcal{X} is taken as $\mathcal{X} = \mathcal{Z}$, and the output space \mathcal{Y} may be defined arbitrarily.

Generative models

The unsupervised learning problem which is most easily framed within the language of statistical learning is that of *generative modelling* where we essentially try to find an approximation $Q(x)$ of the actual distribution $P(x)$. In this case the risk is commonly defined as the Kullback-Liebler divergence between P and Q ,

$$R[Q] \equiv \int \log \frac{P(x)}{Q(x)} dP(x) = D(P||Q). \quad (7.4)$$

Other unsupervised problems include *principal component analysis* (PCA) and *clustering*.

Principal component analysis

Principal component analysis (PCA) is one of the most common methods of dimensional reduction and works by projecting data of high dimension onto the directions of largest variability, the idea being that these directions should also carry most of the relevant information.

Following the presentation of [53] we begin by considering a dataset $\mathcal{D} = (x_1, \dots, x_N)$ where $x_i \in \mathcal{H}^p$ for some Hilbert space \mathcal{H} . We wish to make a dimensional reduction of \mathcal{D} by constructing a rank- q linear model

$$f(\lambda) \equiv \mu + W\lambda, \quad \begin{cases} \mu \in \mathcal{H}^p \\ W \in U(\mathcal{H}^q, \mathcal{H}^p) \\ \lambda \in \mathcal{H}^q \end{cases} \quad (7.5)$$

approximating the data.

Taking the distance function to be the L_2 norm of the error we get the empirical risk

$$R_N[f] = \frac{1}{2} \sum_{i=1}^N \|x_i - \mu - W\lambda_i\|^2. \quad (7.6)$$

One can show that this together with the condition $\sum_i W\lambda_i = 0$ leads to optimal choices of μ and λ_i as

$$\begin{cases} \hat{\mu} = \bar{x} \\ \hat{\lambda}_i = W^\dagger(x_i - \bar{x}). \end{cases} \quad (7.7)$$

It remains to find W . For convenience, assume $\bar{x} = 0$. Then

$$\begin{aligned}
 R_N[f] &= \sum_i x_i^\dagger x_i - 2x_i^\dagger W W^\dagger x_i + x_i^\dagger W W^\dagger W W^\dagger x_i \\
 &= \text{tr } X^\dagger X - 2 \text{tr } X^\dagger W W^\dagger X + \text{tr } X^\dagger W W^\dagger W W^\dagger X \\
 &= \text{tr } X X^\dagger - \text{tr } X X^\dagger W W^\dagger \\
 &= \text{tr } U D U^\dagger - \text{tr } U D U^\dagger W W^\dagger,
 \end{aligned} \tag{7.8}$$

where we have introduced the design matrix $X \equiv (x_1, \dots, x_N)$ and made an eigendecomposition of the *covariance matrix* $\rho \equiv X X^\dagger$ into $U D U^\dagger$, where the diagonal matrix D is ordered such that the largest eigenvalues appear first, i.e. $|D_{11}| \geq |D_{22}| \geq \dots$

Let \tilde{U} and \tilde{D} be the truncation of U and D containing the first q eigenvectors and eigenvalues respectively. Then choosing $W = \tilde{U}$ gives

$$R_N[f] = \text{tr } D - \text{tr } \tilde{D}, \tag{7.9}$$

which is clearly the optimal choice.

The components of $\hat{\lambda}_i$ found by the projection $\hat{\lambda}_i = \tilde{U}^\dagger (x_i - \bar{x})$ are known as the *principal components* of x_i . With a properly chosen rank q , these contain most of the variability of the original dataset and may hence be used as a lower-dimensional substitute for x_i as input to e.g. a supervised learning model.

7.2 Artificial neural networks

Artificial neural networks (ANNs) is a class of machine learning models loosely based on the functionality of biological brains, which have played a large part in the recent surge of interest in ML. There are many different types of ANNs, but common to most is that they represent some parametric function in a distributed manner by modelling interactions between a set of *neurons*, some of which receive input while others produce output.

We will in this thesis consider ANNs which can be described in two main model frameworks – feedforward neural networks (FNNs) which we will introduce presently and Boltzmann machines which are special cases of the Markov random fields to be discussed later in this chapter.

7.2.1 Feedforward neural networks

The simplest, and by far the most popular, type of ANNs are feedforward neural networks. These networks are built upon neurons each of which represent a scalar function $a(x)$ of some input vector x , in which each component is either taken as an external input, or as the output value from another neuron.

The input-output structure of the network may be represented as an open acyclic, directed graph $G = (V, E)$ where each vertex $v \in V$ correspond to a neuron and each edge $(u, v) \in E$ tells us that the output a_u of u is an input to v . An open incoming edge is taken as an input to the network, while an open outgoing edge is an output, so that the network as a whole describes some vector-valued function $f(x)$. Figure 7.1a shows a single neuron represented in this manner, while a full network may be displayed as in figure 7.1b.

The probably most common form of the neuron function $a(x)$ is the McCulloch-Pitts neuron[1] and was introduced already in 1949. It is on the form

$$a(x) = \sigma(w \cdot x + b) \tag{7.10}$$

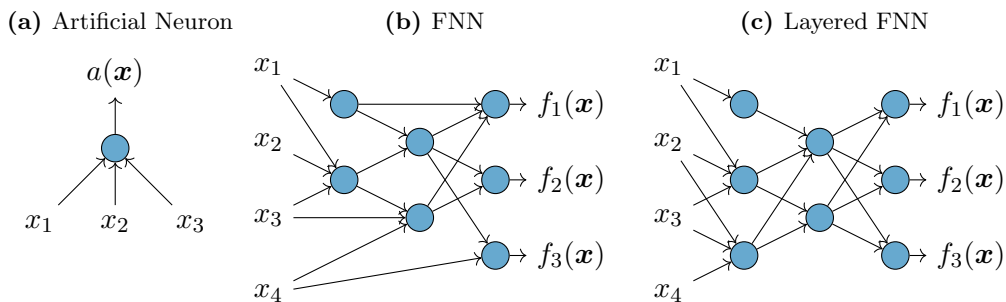


Figure 7.1: Graphical representations of feedforward neural networks. The graph of a single neuron with three inputs is shown in (a), an example of an arbitrary FNN is shown in (b) and (c) shows a layered FNN.

where the *weights* w and the *bias* b are to be determined by the learning algorithm, while the *activation function* $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ is usually taken to be some fixed monotonically increasing function.

A common special case of FNNs is that of figure 7.1c when the neurons are organized in successive layers, such that only the first layer receives external input, each following layer takes the neurons from the directly preceding layer as input and the final layer is treated as the output of the network. This gives a splitting of the $f(x)$ thus described as

$$f(x) \equiv f^{(L)} \circ f^{(L-1)} \circ \dots \circ f^{(1)}(x), \quad (7.11)$$

where

$$f^{(l)}(x) = \bigoplus_i a_i^{(l)}(x), \quad (7.12)$$

with $a_i^{(l)}(x)$ being the output of neuron i in layer l .

For McCulloch-Pitts neurons, this gives the functional form of $f^{(l)}(x)$ as

$$f^{(l)}_i(x) = \sigma \left(\sum_j W_{ij} x_j + b_i \right). \quad (7.13)$$

7.3 Convolutional neural networks

One of the main lessons of theoretical physics is the power of symmetry in understanding and finding efficient descriptions systems. Intuitively, this should carry over to the context of machine learning as well: If the input data for a given machine learning task is known on beforehand to possess some kind of symmetry, it is reasonable to assume that the complexity of the task can be greatly reduced if one is able to hard code this symmetry directly into the algorithm.

For the arguably very common tasks of image- and sound recognition, one such symmetry is translational invariance. One particular class of layered FNNs that have been developed for tasks with translational symmetry is convolutional neural networks (CNNs), which have given rise to some of the greatest successes in ML to date.

7.3.1 Convolutions

A convolution is in its most general form an operation between functions of a real parameter defined as

$$(f * g)(x) \equiv \int d\tau f(\tau)g(x - \tau). \quad (7.14)$$

If $g(x)$ is a probability distribution, one can think of $(f * g)(x)$ as a moving weighted average of $f(x)$. One usually refers to f as the input and g as a kernel [37].

Of more use in a machine learning setting is the discrete convolution which, similarly, is defined as

$$(a * b)_n \equiv \sum_{m=-\infty}^{\infty} a_m b_{n-m}. \quad (7.15)$$

Generalising to functions from \mathbb{N}^N to \mathbb{R} we write

$$(a * b)_n \equiv \sum_{m \in \mathbb{N}^N} a_m b_{n-m}, \quad (7.16)$$

and note that in all the above cases, $a * b = b * a$.

Translational symmetry

Let T be a translation function, defined as $T(a_n) \equiv a_{n+t}$. Then it is easy to see that

$$(T(a) * b)_n = \sum_m T(a_{n-m})b_m = \sum_m a_{n+t-m}b_m = T((a * b)_n), \quad (7.17)$$

i.e. the action of a translation commutes with the action of a convolution.

7.3.2 The structure of a CNN

A CNN is generally composed of a sequence of layers of different types; starting from an input layer, it alternates between *convolutional* and *feature map* layers, interrupted by a *pooling* layer every few iterations and often finishing with one or more layers of a fully connected FNN [55]. Below we will provide a brief explanation of the layer types, taking x as the input and $f(x)$ as the output of each layer.

The input to a layer in a CNN is best described as living on a D -dimensional lattice \mathcal{L} . To be specific, it is given by at each site $s \in \mathcal{L}$ specifying C scalar values, all in all giving a vector $x_s^c \in \mathbb{R}^{C \times M_1 \times \dots \times M_D}$, where M_i is the size of the lattice dimension i . We will refer to C as the number of *channels* of the layer. For the case of RGB images, a standard choice is to take a lattice with $D = 2$ where each site is a pixel, and $C = 3$ with each channel corresponding to the amount of red, green or blue in each pixel.

Convolutional layer

The convolutional layer is the work horse of CNNs and is generally defined as a convolution in the lattice dimensions and an arbitrary affine transformation in the channel dimension, i.e.

$$f^c{}_s(x) = \sum_c (\omega^c{}_c * x^c)_s + b^c = \sum_{c,m} \omega^c{}_{c,m} x^c{}_{s-m} + b^c, \quad (7.18)$$

where b is a bias term and ω is referred to as the kernel of the convolution, usually taken to be non-zero only for a small hypercube of length $K = 1,3,5,7,\dots$ in \mathcal{L} centred around

$\mathbf{m} = 0$. Note also that the number of channels in $f(x)$ is in general different than that in x .

Two subtleties which we should mention before moving on is that to prevent unintended reduction of the lattice size, the input is sometimes padded with zeros before calculating $f(x)$, and that $f(x)$ is not always calculated for every \mathbf{s} , but rather for every S th site in each direction. The number $S \in \mathbb{N}$ is called *stride* and the function calculated by a convolutional layer with stride S and kernel size K may (after some reindexing) be expressed as

$$f^c_{\mathbf{s}}(x) = \sum_c \sum_{\{\mathbf{m}_i\}=1}^K W^c_{c,\mathbf{m}} x_{S(\mathbf{s}-1)+\mathbf{m}} \quad (7.19)$$

Feature map layer

The feature map layers are generally taken as an element-wise application of a non-linear activation function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$, i.e. as functions of the form

$$f(x) \equiv \bigoplus_i \sigma(x_i). \quad (7.20)$$

The activation function is often chosen from the same family of functions as that of the McCulloch-Pitts neuron.

Pooling layer

The pooling layers perform operations to reduce the lattice size in a manner not dissimilar to the course-graining of physical systems in RG methods. Often this is done by partitioning \mathcal{L} into hypercubic blocks B and constructing a new lattice \mathcal{L}' where each site \mathbf{s}' corresponds to a block $B \subset \mathcal{L}$. To put it concretely, the inputs for each block are passed through a function $g : \mathbb{R}^{|B|} \rightarrow \mathbb{R}$ as

$$f^c_{\mathbf{s}'}(x) = g\left(\bigoplus_{\mathbf{s} \in B} x^c_{\mathbf{s}}\right). \quad (7.21)$$

Common choices of $g(\mathbf{x})$ include max pooling where $g(\mathbf{x}) = \max_i x_i$, and average pooling where $g(\mathbf{x}) \equiv \frac{1}{|B|} \sum_i x_i$ [54]. We will also be interested in product pooling, where $g(\mathbf{x}) = \prod_i x_i$.

7.4 Structured probabilistic models

As with the high-dimensional tensors of the tensor networks, it is often helpful for the intuition to represent probability distributions graphically. Models that are represented in this manner are often called *structured probabilistic models* [37], and among these are the Bayesian Networks and Markov Random Fields.

7.4.1 Bayesian networks

The first of the graphical models which we will discuss is the directed graphical model or *Bayesian network*, which represents a probability distribution by factorizing it into conditionals. We can motivate it as follows:



Figure 7.2: Shown in (a) is a Bayesian Network representing the probability distribution $p(x,y,z,w) = p(x)p(y|x)p(z|x,y)p(w|z)$. In (b) is a Markov random field representing the probability distribution $p(x,y,z,w) = \frac{1}{Z}e^{-E_1(x,y,z)-E_2(z,w)}$.

Consider a distribution $p(\mathbf{x})$ over random variables $\mathbf{X} = \{X_1, X_2, \dots, X_N\}$. The chain rule of conditional probabilities gives a factorization of this as

$$p(\mathbf{x}) = p(x_1) \prod_{i=2}^N p(x_i | x_1, \dots, x_{i-1}). \quad (7.22)$$

To simplify the description of $p(\mathbf{x})$ it is desirable to minimize the number of variables to the right of the conditionals, leading us to the following definition:

Definition 7.1. (Markovian parents)

A set $PA_i \subseteq \{X_1, \dots, X_{i-1}\}$ is called Markovian parents of X_i if

$$p(x_i | x_1, \dots, x_{i-1}) = p(x_i | pa_i) \quad (7.23)$$

holds for PA_i , but not for any proper subset of PA_i . The lower case x_i and pa_i are here particular instantiations of the X_i and PA_i .

If we find the Markovian parents of each of the random variables in \mathbf{X} we may represent the full distribution as

$$p(\mathbf{x}) = \prod_i p(x_i | pa_i). \quad (7.24)$$

This description is then easily represented by a directed acyclic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by letting the vertices be $\mathcal{V} = \mathbf{X}$ and letting the parent nodes of node X_i be PA_i ; i.e. $(X_j, X_i) \in \mathcal{E}$ iff $X_j \in PA_i$. A probability distribution represented in this manner is called a *directed graphical model*, and a concrete example is given in figure 7.2a.

7.4.2 Markov random field

Let $G = (V, E)$ be an undirected graph, associate with each vertex $v \in V$ a random variable X_v and define $N(v)$ as the set of vertices adjacent to v . Let x_v be a particular configuration of X_v and, similarly, x_A be a configuration of all X_v such that $v \in A \subseteq V$. Let $p(x)$, where $x \equiv x_V$ be the distribution over all random variables of the graph. Then $p(x)$ is said to be a *Markov random field*[56] if for all $v \in V$,

$$p(x_v | x_{V \setminus \{v\}}) = p(x_v | x_{N(v)}), \quad (7.25)$$

that is, if all x_v are only dependent on their neighbours.

As shown in e.g. [56], a Markov random field can always be expressed in the form of a nearest neighbour Gibbs distribution

$$p(x) = \frac{1}{Z} e^{-E(x)}, \quad (7.26)$$

where $Z \equiv \sum_x e^{-E(x)}$ is a normalizing factor known as a *partition function* and the potential $E(x)$ may be expanded as a sum of clique potentials $E_C(x_C)$ on the form

$$E(x) = \sum_{C \in \mathcal{C}} E_C(x_C), \quad (7.27)$$

with \mathcal{C} being the set of cliques in G . Conversely, any nearest neighbour Gibbs distribution is a Markov random field on the same graph.

An example of a Markov random field with nearest neighbour Gibbs distribution is shown in figure 7.2b.

7.5 Boltzmann machines

A Boltzmann machine is a model for unsupervised learning of a true probability distribution $p_{\text{data}}(v)$ over a binary vector $v \in \{0,1\}^n$, called the *visible variables* of the model.

The probability distribution of the model $p(v)$ is represented as a marginal distribution of a Markov random field $p(s)$, where $s = (v, h)$ and $h \in \{0,1\}^m$ is a set of *hidden variables* which are introduced to give $p(s)$ higher representational power, and where the Gibbs potential is on the form

$$E(s) = -\langle Ws, s \rangle - \langle s, a \rangle. \quad (7.28)$$

Here $a \in \mathbb{R}^{n+m}$ and $W = W^T$ is a linear map from \mathbb{R}^{n+m} to itself which is zero on the diagonal. The desired distribution over v is then arrived at by summing over all configurations of the hidden variables, i.e.

$$p(v) = \sum_h p(s). \quad (7.29)$$

The graph representation of $p(s)$ is easily seen to be $G = (V, E)$ such that $V = s$ and $(s_i, s_j) \in E$ iff $W_{ij} \neq 0$, since this gives non-zero terms in (7.28) only for cliques in G . As an example, the Boltzmann machine corresponding to figure 7.2b has W_{xy} , W_{xz} , W_{yz} and W_{zw} (and their transposes) as the only non-zero weights and if we consider x and y to be the visible variables, the corresponding distribution is of course

$$p(x, y) = \sum_{z, w} \frac{1}{Z} e^{-E_1(x, y, z) - E_2(z, w)} = \sum_{z, w} \frac{1}{Z} e^{2W_{xy}xy + 2W_{xz}xz + 2W_{yz}yz + 2W_{zw}zw + \langle s, a \rangle}. \quad (7.30)$$

The objective is then to find the weights W and biases a which minimizes the difference between the model distribution $p(v)$ and the actual distribution $p_{\text{data}}(v)$, e.g. by minimizing the KL divergence $D(p_{\text{data}} \| p)$. Doing this for a general Boltzmann machine quickly becomes a very difficult task as one tries to increase the number of variables; the probability space grows exponentially in the number of variables and sampling from the resulting distributions is a research field in its own right. Because of this, one generally constrains the connectivity of the network in some manner when doing ML with Boltzmann Machines. Some of the most common methods of doing so will be discussed in the upcoming sections.

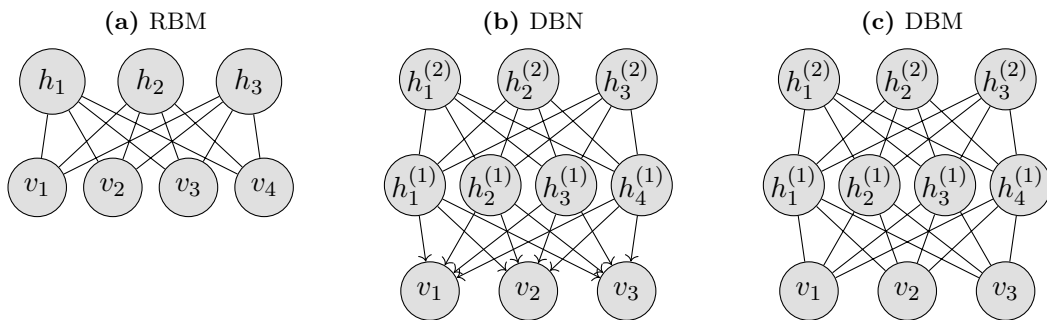


Figure 7.3: Panels (a), (b) and (c) shows graphical representations of a small dense restricted Boltzmann machine, deep belief network and deep Boltzmann machine respectively. The RBM and DBM are represented as Markov random fields, while the DBN is represented as a Markov random field in the two uppermost layers but a Bayesian network in all lower layers.

7.5.1 Restricted Boltzmann machines

A Restricted Boltzmann machine (RBM) is a Boltzmann Machine with a set of visible units v , and hidden units h where there is no intralayer connections in the corresponding graph, i.e. the graph is on the form of figure 7.3a. This immediately gives that the conditional probabilities follow

$$p(v|h) = \prod_i p(v_i|h), \quad \text{and} \quad p(h|v) = \prod_i p(h_i|v), \quad (7.31)$$

and that the Gibbs energy may be written

$$E(v, h) = -\langle Wv, h \rangle - \langle v, a \rangle - \langle b, h \rangle, \quad (7.32)$$

where W is a linear map from \mathbb{R}^n to \mathbb{R}^m , $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$.

The benefit of restricting the connectivity in this manner is that it renders both the v_i independent given h and the h_i independent given v , which allows for highly efficient block Gibbs Sampling.

We can also simplify the conditional distributions $p(v_i|h)$, using that $v_i \in \{0,1\}$, and $\sum_{v_i} p(v_i|h) = 1$, giving

$$p(v_i = 1|h) = \frac{1}{1 + e^{-\sum_k h_k W_{ki} - a_i}} = \sigma\left(\sum_k h_k W_{ki} + a_i\right), \quad (7.33)$$

where $\sigma(x) \equiv (1 + e^{-x})^{-1}$ is known as the *sigmoid function*, which incidentally is also one of the most common choices of nonlinearity in FNNs. By symmetry for $p(h_i = 1|v)$ we get

$$p(h_i = 1|v) = \sigma\left(\sum_k W_{ik} v_k + b_i\right). \quad (7.34)$$

7.5.2 Deep belief networks

Deep belief networks (DBN) are one of several methods of extending RBMs to deep architectures. It was one of the first non-convolutional deep networks to be successfully

trained[8], but has since largely fallen out of favour[37]. They are nevertheless conceptually very interesting, particularly in that the weights learned by a DBN have been shown to be very suitable as initial conditions for FNNs employed for classification[57].

A DBN with l hidden layers is in its simplest form a generative model which represents a probability distribution over the visible variables $v_i \in \{0,1\}$ and hidden variables $h^{(j)}_i \in \{0,1\}$ where $j = 1, \dots, l$ is the index of the layer. The probability distribution is represented as a mixture between a directed and an undirected model, with

$$\begin{aligned} p(h^{(l)}, h^{(l-1)}) &\propto e^{\langle h^{(l)}, b^{(l)} \rangle + \langle h^{(l-1)}, b^{(l-1)} \rangle + \langle W h^{(l)}, h^{(l-1)} \rangle}, \\ p(h^{(j)}_i = 1 | h^{(j+1)}) &= \sigma \left(b^{(j)}_i + (W^{(j+1)} h^{(j+1)})_i \right), \quad j = 1, \dots, l-2 \\ p(v_i = 1 | h^{(1)}) &= \sigma \left(a_i + (W^{(1)} h^{(1)})_i \right), \end{aligned} \quad (7.35)$$

where $\sigma(x) \equiv (1 + e^{-x})^{-1}$ is the sigmoid function. An example of a small DBN with $l = 1$ is shown in figure 7.3b. Drawing a sample from a DBN is done by first drawing a sample from the RBM defined by the top two layers, after which one can use ancestral sampling to draw a sample for the rest of the variables.

The training of a DBN is typically done greedily by stacking RBMs; one starts by training the RBM with variables $v, h^{(1)}$ and parameters $a, W^{(1)}, b^{(1)}$ to minimize the KL divergence $D(p_{\text{data}} \| p^{(1)})$ where $p^{(1)}(v)$ is the marginal distribution over v modelled by the RBM. Once this training is done, one proceeds to the next layer by essentially defining a “true” distribution over $h^{(1)}$ as

$$p_{\text{data}}^{(1)}(h^{(1)}) \equiv \sum_v p^{(1)}(h^{(1)} | v) p_{\text{data}}(v) \quad (7.36)$$

where $p^{(1)}(h^{(1)} | v)$ is the conditional distribution defined by the RBM. The following layer is then trained as an RBM with variables $h^{(1)}, h^{(2)}$ and parameters $b^{(1)}, W^{(2)}, b^{(2)}$ this time minimizing the KL divergence $D(p_{\text{data}}^{(1)} \| p^{(2)}(h^{(1)}))$ where $p^{(2)}$ is the marginal distribution over $h^{(1)}$ defined by the second RBM.

This process of stacking RBMs can be repeated indefinitely and is motivated by that each step decreases the lower bound on $D(p_{\text{data}} \| p)$ where $p(v)$ here is the marginal distribution over v of the full model.

7.5.3 Deep Boltzmann machines

Another extension of the RBM is the deep Boltzmann machine (DBM) [58], which is a Boltzmann machine where the hidden variables are partitioned into l layers, $h^{(1)}, h^{(2)}, \dots, h^{(l)}$, and the corresponding graph is restricted to only contain edges between neighbouring layers as shown for $l = 2$ in figure 7.3c. Defining $h^{(0)} \equiv v$ for convenience this puts the Gibbs energy on the form of

$$E(\{h^{(j)}\}) = - \sum_{i=0}^l \left(\langle W^{(i+1)} h^{(i)}, h^{(i+1)} \rangle + \langle h^{(i)}, b^{(i)} \rangle \right). \quad (7.37)$$

This construction renders odd layers conditionally independent of even layers, which allows for efficient block Gibbs sampling in much the same way as for RBMs.

7.6 Kernel learning

Kernel learning is an area of ML with roots in statistical learning theory, which may in many cases be thought of as a method of increasing the expressive power of linear models

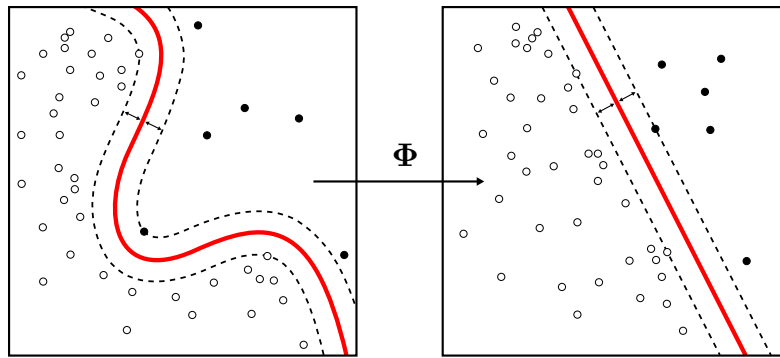


Figure 7.4: A map $\Phi(x)$ taking the input patterns $x_i \in \mathcal{X}$ (left panel) with $y_i = 1$ (filled circles) or -1 (open circles) to an inner product space \mathcal{H} (right panel) where the classes are linearly separable.¹

by first non-linearly mapping the data into a higher-dimensional inner product space \mathcal{H} , called a *feature space*, although the actual computations can often be carried out without reference to this space owing to a neat construct called the *kernel trick* (hence the name “kernel learning”). The theory of kernel learning is surprisingly elegant and diverse, and a good review, from which much of the following material is picked, can be found in [59].

Although many algorithms making use of inner product spaces, such as PCA, can be easily put into a kernel learning context, it is natural to give an introduction to the field by first considering its most famous algorithm – the Support Vector Machine (SVM).

7.6.1 Support vector machines

We start from the supervised learning problem of constructing a classifier between two categories. Consider a finite set of empirical labelled data $\mathcal{D} = \{(x_i, y_i); i = 1, \dots, m\}$, with samples $x_i \in \mathcal{X}$ and labels $y_i \in \{-1, 1\}$, which are iid and drawn from a distribution $P(x, y)$ and where \mathcal{X} is an arbitrary set.

Our goal is to based on our empirical data find a function

$$f : \mathcal{X} \rightarrow \{-1, 1\} \quad (7.38)$$

which minimizes the risk

$$R[f] \equiv \frac{1}{2} \int |f(x) - y| dP(x, y), \quad (7.39)$$

here using the L_1 norm as distance function.

The kernel learning approach to solving this problem is to introduce a so-called *feature map* $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ to some inner product feature space \mathcal{H} , the idea being that for a sufficiently high dimension of \mathcal{H} and appropriately chosen Φ , there will tend to exist a hyperplane in \mathcal{H} separating the classes, in which case the problem is said to be *linearly separable* (see fig. 7.4).

Such a hyperplane P may be parametrized as

$$P_{\mathbf{w}, b} \equiv \{\mathbf{x} \in \mathcal{H} : \langle \mathbf{w}, \mathbf{x} \rangle + b = 0\}, \quad (7.40)$$

¹Image adapted from https://commons.wikimedia.org/wiki/File:Kernel_Machine.svg

where $\mathbf{w} \in \mathcal{H}$ is its normal and $b \in \mathbb{R}$ is its distance to the origin, and we may construct a corresponding decision function as

$$f(x) \equiv \text{sgn} \left(\langle \mathbf{w}, \Phi(x) \rangle + b \right). \quad (7.41)$$

It turns out that there exists a unique optimal separating hyperplane, which maximizes the distance to the closest data points, meaning that the corresponding \mathbf{w} and b are determined by

$$\text{argmax}_{\mathbf{w}, b} \min_{\substack{\mathbf{x} \in P_{\mathbf{w}, b} \\ i \in \{1, \dots, m\}}} \|\mathbf{x} - \mathbf{x}_i\|, \quad (7.42)$$

where $\mathbf{x}_i \equiv \Phi(x_i)$ are the *feature vectors* for the training data x_i and $\|\mathbf{x}\| \equiv \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$. Selecting the optimal hyperplane is expected to be beneficial to generalization, since we expect new samples to likely be mapped to points close to existing samples in the same class.

An essentially equivalent way of expressing the optimal hyperplane is as the pair \mathbf{w} , b which minimizes

$$\frac{1}{2} \|\mathbf{w}\|^2, \quad (7.43)$$

subject to the constraints

$$y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \quad \forall i \in \{1, \dots, m\}. \quad (7.44)$$

Employing the Lagrangian multiplier method this can be expressed as finding the saddle point of the Lagrangian

$$L(\mathbf{w}, b, \lambda) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \lambda_i (y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + y_i b - 1), \quad (7.45)$$

which is a minimum in variables \mathbf{w} and b but a maximum in the nonnegative Lagrangian multipliers λ_i . At the saddle point, the partial derivatives with respect to b and \mathbf{w} vanish, giving in order

$$0 = \frac{\partial L}{\partial b} = \sum_{i=1}^m \lambda_i y_i, \quad (7.46)$$

$$0 = \frac{\partial L}{\partial \mathbf{w}} = \mathbf{w}^\dagger - \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \quad \implies \quad \mathbf{w} = \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i^\dagger. \quad (7.47)$$

For those (x_i, y_i) where (7.44) is not an equality, one can show² that the corresponding λ_i is necessarily zero, so that

$$\lambda_i (y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + y_i b - 1) = 0 \quad \forall i. \quad (7.48)$$

This tells us that the optimal \mathbf{w} is a linear combination of the feature space representations \mathbf{x}_i of the training samples with non-zero λ_i . These \mathbf{x}_i are called *support vectors*, and correspond to the points that intersect the dashed lines in fig. 7.4. The non-zero λ_i can now be determined by solving the *dual optimization problem* of maximizing

$$\sum_i \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle; \quad \lambda_i \geq 0 \quad (7.49)$$

²This is known from optimization theory as the Karush-Kuhn-Tucker complementarity conditions

which is arrived at by substitution of (7.46) and (7.47) into the original Lagrangian. After the λ_i are found, b is easily determined from (7.48), and our objective function can be put on the form

$$f(x) = \text{sgn} \left(\sum_{i=1}^m \lambda_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right). \quad (7.50)$$

There is an issue with this construction, however, in that the feature space for real world applications often needs to be of sufficiently high dimension to make it impractical to represent directly on a computer (although tensor network methods might be able to increase the dimensions which are directly tractable significantly, as we will see in the next chapter). Fortunately, this can be helped by means of the *kernel trick*, where one simply substitutes the inner products with the corresponding kernel functions

$$k(x, x') \equiv \langle \Phi(x), \Phi(x') \rangle. \quad (7.51)$$

One should note that this is possible due to the fact that we were able to express \mathbf{w} entirely in terms of inner products between feature vectors of known input patterns. This substitution gives

$$f(x) = \text{sgn} \left(\sum_i \lambda_i y_i k(x, x_i) + b \right), \quad (7.52)$$

with the λ_i determined as those that maximize

$$\sum_i \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j k(x_i, x_j), \quad (7.53)$$

under the constraints that all $\lambda_i \geq 0$ and $\sum_i \lambda_i y_i = 0$.

This model, which can be elaborated on to e.g. find optimal hyperplanes for problems which are not quite linearly separable (which is perhaps a bit more realistic), or include more categories into the classification, is what is known as an SVM. The method of mapping input patterns into high-dimensional inner product spaces and using the kernel trick to make computations tractable is, however, much more general and forms the core of the subject of kernel learning.

Part III

Quantum theory for machine learning

8

Tensor network learning

Having gone through the theoretical foundations, we are now ready to start looking at the main subject of the thesis, which is to study the applications of quantum many-body theory to machine learning. We will go through a number of different approaches to this, but common to all of them is that the training data is mapped to vectors in tensor product Hilbert spaces, allowing the use of numerical and theoretical tools from QM to construct and understand ML models.

Nearly all [17–22, 24, 60–62] of these may be expressed as linear maps acting on said Hilbert spaces, making tensor networks an ubiquitous tool in the context. This in turn warrants the use of Tensor Network Learning (TNL) as a term for the subject, for lack of a better name.

8.1 Tensor product Hilbert spaces in machine learning

Consider the case of wanting to learn some function $f(\mathbf{x})$ when the input \mathbf{x} has a tuple structure, i.e. when $\mathbf{x} \in \mathcal{X}^M$, or equivalently $\mathbf{x} = (x_1, \dots, x_M)$ with $x_i \in \mathcal{X}$. As simple examples we may consider \mathbf{x} to be a time series of temperature readings, where each x_i is a temperature so that $\mathcal{X} = \mathbb{R}$, or it might be an RGB-image where each x_i represents the intensities of red, green and blue in a pixel in which case we could have $\mathcal{X} = [0, 1]^3$.

8.1.1 Square-integrable functions

To make progress, we will inevitably need to restrict the class of functions we are interested in, and a good place to start is to restrict \mathcal{X} to being a measurable space and consider square-integrable functions from \mathcal{X}^M to some Hilbert space \mathcal{H}_{out} , i.e. $f \in L^2(\mathcal{X}^M; \mathcal{H}_{\text{out}})$.

If $L^2(\mathcal{X})$ is separable, it is a standard result that $L^2(\mathcal{X}^M) \cong L^2(\mathcal{X})^{\otimes M}$. Hence, if $\{\phi_i\}_i$ is a basis for $L^2(\mathcal{X})$, the set of all products $\{\prod_{j=1}^M \phi_{i_j}\}_{\{i_k\}}$ forms a basis for $\mathcal{H} \equiv L^2(\mathcal{X}^M)$ and we may write an arbitrary element in \mathcal{H} as

$$g(\mathbf{x}) = \sum_{\{i_k\}} T_{i_1, i_2, \dots, i_M} \phi_{i_1}(x_1) \phi_{i_2}(x_2) \dots \phi_{i_M}(x_M) = T \cdot \Phi(\mathbf{x}), \quad (8.1)$$

where

$$\Phi(\mathbf{x}) \equiv \bigotimes_i \phi(x_i). \quad (8.2)$$

We may hence write our sought function f as

$$f(\mathbf{x}) = W \cdot \Phi(\mathbf{x}), \quad (8.3)$$

where $W \in \text{hom}(\mathcal{H}, \mathcal{H}_{\text{out}})$, and we may rephrase our learning problem as the problem of finding the optimal W .

Of course, to perform actual ML, we will in practice need to represent W on a computer. The extent to which this is possible will of course depend on the choice of \mathcal{X} , and it is hence interesting to consider two special cases.

The simplest of these is the case where \mathcal{X} is a finite set, i.e. $\mathcal{X} = \{x_1, \dots, x_n\}$. It is easily seen that a basis over $L^2(\mathcal{X})$ is given by $\phi_i(x) = \delta_{x_i, x}$, with $\delta_{x_i, x}$ being the Kronecker delta. This tells us that $\dim L^2(\mathcal{X}) = |\mathcal{X}|$, and hence $\dim \mathcal{H} = |\mathcal{X}|^M$, which is feasible to handle numerically provided $|\mathcal{X}|$ and M are sufficiently small.

The other case we will consider is when the dimension of $L^2(\mathcal{X})$ is countably infinite, as is the case when e.g. \mathcal{X} is \mathbb{R} , \mathbb{C} or an interval in \mathbb{R} . In this case, the dimension of \mathcal{H} is of course also countably infinite, making the full representation of arbitrary W impossible. However we can still hope to capture a reasonably large class of functions by choosing an appropriate basis $\{\phi_i\}$ for $L^2(\mathcal{X})$ and truncating it at $i = d$ for some $d < \infty$. Repeating the analysis above, this leads to a total dimension of d^M for the resulting tensor product Hilbert space.

8.1.2 Local feature maps

Another way of arriving at the above construction is to position ourselves in a kernel learning context, in which case the starting point is to construct a feature map

$$\Phi : \mathcal{X}^M \rightarrow \mathcal{H}, \quad (8.4)$$

where \mathcal{H} is some inner product space.

The most common choice for data with tuple-structure is to construct the feature space map as the M th-fold direct sum of *local feature maps*

$$\phi : \mathcal{X} \rightarrow \mathcal{H}_\phi, \quad (8.5)$$

where \mathcal{H}_ϕ is some arbitrary inner product space. This gives the full feature map as

$$\Phi(\mathbf{x}) = \bigoplus_i \phi(x_i), \quad (8.6)$$

and hence $\mathcal{H}_\oplus = \mathcal{H}_\phi^M$.

There is another equally simple choice, however, which also preserves the tuple structure but has the potential of creating a much more rich family of features. That is to instead take the feature space map as the M th-fold tensor product of the local feature maps, i.e. let

$$\Phi(\mathbf{x}) = \bigotimes_i \phi(x_i), \quad (8.7)$$

giving $\mathcal{H} = \mathcal{H}_\phi^{\otimes M}$.

8.1.3 Connecting to tensor networks

Employing tensor product Hilbert spaces in ML is interesting for a number of reasons. First, the components of a vector in \mathcal{H} are given by products of components of vectors in \mathcal{H}_ϕ , making it possible to have linear models which can respond very richly to correlations in the data. Since linear algebra is well understood, models developed in this manner have a good hope of being possible to fully understand, in contrast to the arguably more mysterious deep neural networks.

Second, the tensor product structure of \mathcal{H} puts $\Phi(\mathbf{x})$ on an identical form to the wave function of an M -body quantum mechanical product state. It is therefore natural to

8.2.1 Gradient descent

The simplest learning procedure to consider is that of gradient descent, where each tensor T_v is updated in turn according to

$$T_v \rightarrow T_v + \Delta T_v, \quad (8.13)$$

where

$$\Delta T_v \equiv -\eta \frac{\partial R[f]}{\partial T_v} = -\eta \sum_i \frac{\partial R[f]}{\partial f(\mathbf{x}_i)} \frac{\partial f(\mathbf{x}_i)}{\partial T_v} = -\eta \sum_i \frac{\partial R[f]}{\partial f(\mathbf{x}_i)} W_{T_v \mathbf{x}_i}, \quad (8.14)$$

with $W_{T_v} \equiv \partial W / \partial T_v$ being the environment of tensor T_v and $\eta > 0$ the step length. Given that $\partial R[f] / \partial f(\mathbf{x}_i)$ is sufficiently simple, this should be a relatively contractible computation, and it would be interesting to see what could be achieved by applying gradient descent, or any of its standard extensions to tensor network models.

8.2.2 Generalized DMRG

One extension of gradient descent which is special to tensor networks is that which is commonly used within DMRG, where the gradient descent is performed with respect to blocks of tensors. Generalizing to arbitrary TNs gives the following algorithm:

Each step of the procedure begins by selecting an edge $(u, v) \in \bar{E}$. The goal of the step is then to optimize the corresponding tensors T_u and T_v , which we do by first contracting them along their shared edges, forming $B \equiv T_u T_v$. The second step is to perform gradient descent as above with respect to B , i.e. let $B \rightarrow B + \Delta B$ where

$$\Delta B = -\eta \sum_i \frac{\partial R[f]}{\partial f(\mathbf{x}_i)} W_{B \mathbf{x}_i}, \quad (8.15)$$

where W_B is the environment of B .

Once an optimal B is found, we then retrieve updated tensors T'_u, T'_v by first performing an SVD

$$B = U^\dagger D V \approx \tilde{U}^\dagger \tilde{D} \tilde{V}, \quad (8.16)$$

truncating it to the m eigenvalues with biggest absolute values and then letting

$$T'_u \equiv \tilde{U}^\dagger \sqrt{\tilde{D}} \quad \text{and} \quad T'_v \equiv \sqrt{\tilde{D}} \tilde{V}, \quad (8.17)$$

where \tilde{D} may be distributed arbitrarily between the tensors. The number of kept eigenvalues m may be kept fixed or be selected such that the truncation error is less than some tolerance ϵ . The second case is very interesting to consider, since m is the new bond dimension between u and v , and allowing it to be dynamically updated essentially allows the expressive power of $f(\mathbf{x})$ to change dynamically in a controlled manner. This is a quite rare feature among machine learning models and may be seen as one of the main motivations for studying TNML.

8.2.3 Supervised learning with MPS

A concrete example of the DMRG algorithm being applied to classification was considered by Stoudenmire & Schwab [17] and proceeds as follows. Consider the task of learning to classify samples $\mathbf{x} \in \mathcal{H} = \otimes_j \mathcal{H}_j$ into d classes based on training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ where y_i is the class of sample i . We take $y_i \in \mathbb{R}^d$ to be encoded as a one-hot vector,

meaning that $y_i^l = \delta_{i_i}^l$, where $l_i \in 1, 2, \dots, d$, and define our classifier as the $f(\mathbf{x})$ of eq. (8.12), with $\mathcal{H}_{\text{out}} = \mathbb{R}^d$, and W restricted to an extension of an MPS of the form

$$W^l_{\mathbf{s}} \sim \begin{array}{c} \\ | \\ \bullet \\ | \\ s_1 \end{array} \begin{array}{c} \\ | \\ \bullet \\ | \\ s_2 \end{array} \begin{array}{c} l \\ | \\ \bullet \\ | \\ s_3 \end{array} \begin{array}{c} \\ | \\ \bullet \\ | \\ s_4 \end{array} \begin{array}{c} \\ | \\ \bullet \\ | \\ s_5 \end{array} . \quad (8.18)$$

We can then perform the block-wise gradient descent of DMRG as discussed above, with the empirical risk function e.g. chosen to be the quadratic cost

$$R[f] \equiv \frac{1}{2} \sum_i (f(\mathbf{x}_i) - y_i)^2, \quad (8.19)$$

giving the gradient descent update as

$$\Delta B \equiv -\eta \frac{\partial R[f]}{\partial B} = \eta \sum_i (y_i - f(\mathbf{x}_i)) W_B \mathbf{x}_i. \quad (8.20)$$

Stoudenmire & Schwab applied this algorithm to the MNIST data set, consisting of 60000 labelled 28×28 pixel (although they down-sample to 14×14) grey-scale images of handwritten digits, using the local feature map $\Phi(\mathbf{x}) = \otimes_i \phi(x_i)$, where $x_i \in [0, 1]$ is the darkness of pixel i and

$$\phi(x) = \begin{pmatrix} \cos\left(\frac{\pi}{2}x\right) \\ \sin\left(\frac{\pi}{2}x\right) \end{pmatrix}. \quad (8.21)$$

They run the algorithm with a cap on the MPS bond dimensions to less than m achieving test error rates down to less than 1% for $m = 120$. This is a great proof of concept for the method, but to get an appreciation of its performance compared to state-of-the-art methods, it would be very interesting to apply it to more difficult datasets.

8.3 Convolutional arithmetic circuits

Convolutional Arithmetic Circuits (CACs) are a special and somewhat unusual class of FNNs bearing significant resemblance to convolutional networks and introduced by Cohen et al [63]. The model has the notable feature of only needing the mathematical operations of products and sums in all but the first layer – hence the name Arithmetic Circuit. It is also quite suitable for theoretical analysis, as they show e.g. by tying it to the hierarchical Tucker decompositions to explain the depth efficiency of the model.

We are, however, for this thesis more interested in the connection made to tensor networks by Levine et al [24, 64], and the following analysis on quantum entropy as a measure of which correlations can be modelled by the network.

Although the definition of what constitutes a CAC seems to have evolved and diverged somewhat over time, they can (as far as we are able to tell) most often be seen as special cases of networks on the following form, presented in [64].

A CAC calculates a function

$$h : \mathcal{H}_{\text{in}}^N \rightarrow \mathcal{H}_{\text{out}} \quad (8.22)$$

where \mathcal{H}_{in} and \mathcal{H}_{out} are Hilbert spaces of finite dimension, as a sequence of layers;

$$h(x) \equiv f^L \circ f^{L-1} \circ \dots \circ f^1 \circ f^0(x). \quad (8.23)$$

The first layer which takes $x \in \mathcal{H}_{\text{in}}^N$ as input, is a local feature map

$$\phi : \mathcal{H}_{\text{in}} \rightarrow \mathcal{H}, \quad (8.24)$$

where \mathcal{H} is a finite-dimensional Hilbert space with $\dim \mathcal{H} = d$. This ϕ acts on each component $x_i \in \mathcal{H}_{\text{in}}$ of x , giving the output of the layer as

$$f^0(x) = \bigoplus_i \phi(x_i). \quad (8.25)$$

This initial layer is followed by a series of layers which may be thought of as mixtures between convolutions and product pooling, and may be defined (omitting the layer index for convenience) as

$$f_s^c(x) = \prod_{m=1}^K \sum_{c'} \omega_m^{c'} x_{S(s-1)+m}^{c'} \quad (8.26)$$

where S is the stride and K is the kernel size. Note also that if we take ω to be independent of m , the above is equivalent to a convolution with a kernel of length 1 and stride 1 followed by a product pooling with window size K and stride S . The number of outputs from a layer of this form is (assuming no padding is used)

$$N_{\text{out}} = \left\lfloor \frac{N - K}{S} + 1 \right\rfloor. \quad (8.27)$$

Layers on this form are repeated until the lattice is reduced to a single site, after which the final layer is taken as a linear function $f^L(x) = W^L \cdot x$. It is of interest to count the number of outputs of a given layer under this scheme. Denoting the kernel size, stride and number of outputs of layer l with K_l , S_l and N_l respectively and defining $N_0 \equiv N$, the equation (8.28) defines the difference equation

$$N_l = \left\lfloor \frac{N_{l-1} - K_l}{S_l} + 1 \right\rfloor. \quad (8.28)$$

The solution for the case $K_l = S_l \forall l$ becomes particularly simple, and assuming N is such that no rounding is performed, we find

$$N_l \Big|_{K_i=S_i \forall i} = N \prod_{i=1}^{l-1} \frac{1}{K_i}. \quad (8.29)$$

8.3.1 Tensor network formulation

One of the central findings of [24, 64] is that the convolutional layers $f_s^c(x)$ are equivalent to a tensor contraction between the $(K+1)$ -mode tensor

$$W^c_{c'_1, c'_2, \dots, c'_K} \equiv \sum_{\{c_i\}} \delta^c_{c_1, c_2, \dots, c_K} \omega_1^{c_1} \omega_2^{c_2} \dots \omega_K^{c_K} c'_K = \begin{array}{c} c \\ \triangle \\ \omega_1 \quad \omega_2 \quad \dots \quad \omega_K \\ | \quad | \quad \dots \quad | \\ c'_1 \quad c'_2 \quad \dots \quad c'_K \end{array} = \begin{array}{c} | \\ \bigcirc W \\ | \quad \dots \end{array}, \quad (8.30)$$

with $\delta^c_{c_1, \dots, c_K}$ being the Kronecker delta, and the K -fold tensor product

$$(x_{S(s-1)+1} \otimes x_{S(s-1)+2} \otimes \dots \otimes x_{S(s-1)+K})^{c'_1, \dots, c'_K}. \quad (8.31)$$

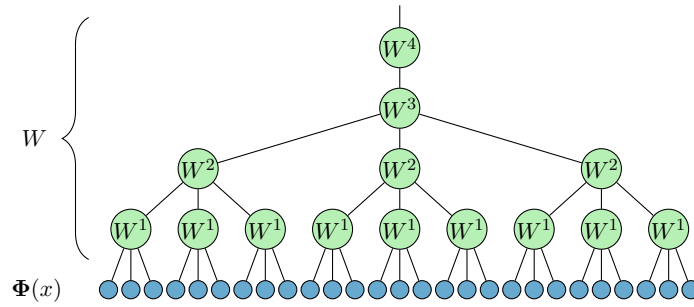


Figure 8.1: The tensor network representation of $h(x)$ for a 1D convolutional arithmetic circuit with four layers and $S = K = 3$. Here W^1, W^2, W^3 have the internal structure shown in (8.30), while W^4 may be any matrix of appropriate dimensionality.

Since this is true for each layer, we may ask ourselves if it is possible to express the entire $h(x)$ as tensor contractions. That is, can we put $h(x)$ on the form of a tensor network? In general this is not possible, since with $K > S$, we will end up with $f_s^c(x)$ being dependent on the same $x_{s'}$ for several values of s , leading to terms proportional to $x_{s'}^2$ in the following layers, which is not possible to express with a TN acting on $\bigotimes_{s'} x_{s'}$.

However, restricting to $K = S$, no such terms appear and we get an overall function

$$h(x) = W \cdot \Phi(x), \quad (8.32)$$

where

$$\Phi(x) \equiv \bigotimes_i \phi(x_i) \quad (8.33)$$

and

$$W = W^L \cdot W^{L-1} \cdot (W^{L-2})^{\otimes N_{L-2}} \cdot \dots \cdot (W^1)^{\otimes N_1} \quad (8.34)$$

with each W^l with $l < L$ on the form of (8.30) and the final W^L as an ordinary matrix with appropriate dimensionality.

As an example, the resulting TN for a four-layer CAC with $K = S = 3$ is shown in figure 8.1. Note especially that the overall function (8.32) together with the feature map (8.33) puts this special case of the CAC model firmly within the framework of TNL.

8.3.2 nCACs

A significant note to make when discussing CACs is that some of the theoretical results from e.g. [24] only hold in their strongest form if one removes the translational symmetry in the convolutional layers, i.e. if (8.35) is altered to

$$f_s^c(x) = \prod_{m=1}^K \sum_{c'} \omega_{m,s}^{c'} x_{S(s-1)+m}^{c'}. \quad (8.35)$$

Note that the only difference is that we now allow different matrices ω to act on each input x_i , in effect taking $W^l \rightarrow W_s^l$ in figure 8.1. To avoid confusion, we will use the term nCAC (for either *near-convolutional arithmetic circuit* or *non-convolutional arithmetic circuit*, depending on the preferences of the reader) for this kind of relaxation of a CAC.

8.4 Unsupervised coarse graining

A very interesting algorithm in this context is the one named *unsupervised coarse graining*, which was proposed by Stoudenmire[18]. It essentially performs a kernel PCA in the feature space by using an adaptation of the entanglement renormalization procedure by Vidal[16] to approximately diagonalize the feature space covariance matrix.

8.4.1 The algorithm

Start by considering a set of training patterns \mathbf{x}_i , $i = 1, \dots, N$, and denote their images in feature space by \mathbf{x}_i^s , i.e. define

$$\mathbf{x}_i^s \equiv \Phi^s(\mathbf{x}_i) = \phi^{s_1}(x_{i,1})\phi^{s_2}(x_{i,2}) \dots \phi^{s_M}(x_{i,M}) \quad (8.36)$$

where $x_{i,j}$ is the j th component of \mathbf{x}_i and $\Phi(\mathbf{x}) \equiv \bigotimes_i \phi(x_i)$ is a local feature map. We may represent this in TN notation as

$$\mathbf{x}_i^s = \begin{array}{c} s_1 \quad s_2 \quad s_3 \quad \dots \quad s_M \\ | \quad | \quad | \quad \dots \quad | \\ \bullet \quad \bullet \quad \bullet \quad \dots \quad \bullet \end{array} = \begin{array}{c} s_1 \quad s_2 \quad s_3 \quad \dots \quad s_M \\ | \quad | \quad | \quad \dots \quad | \\ \text{---} \text{---} \text{---} \text{---} \text{---} \text{---} \\ | \\ i \end{array}, \quad (8.37)$$

noting that including i as a tensor index removes the tensor product structure of the vector.

The feature space covariance matrix is

$$\rho \equiv \frac{1}{N} \sum_i \mathbf{x}_i \mathbf{x}_i^\dagger = \frac{1}{N} \sum_i \begin{array}{c} \bullet \quad \bullet \quad \bullet \quad \dots \quad \bullet \\ | \quad | \quad | \quad \dots \quad | \\ \bullet \quad \bullet \quad \bullet \quad \dots \quad \bullet \end{array} = \begin{array}{c} \text{---} \text{---} \text{---} \text{---} \text{---} \text{---} \\ | \quad | \quad | \quad \dots \quad | \\ \text{---} \text{---} \text{---} \text{---} \text{---} \text{---} \\ | \\ \dots \end{array} \quad (8.38)$$

and the goal of Kernel PCA is then to find the eigenspace decomposition

$$\rho = U^\dagger \Lambda U, \quad (8.39)$$

where U is a unitary map on \mathcal{H} and $\Lambda = \text{diag}\{\lambda_1, \lambda_2, \dots\}$ with $\lambda_1 \geq \lambda_2 \geq \dots$ being the eigenvalues of ρ . It is of course unfeasible to perform this computation exactly for almost any realistic problem, but we may actually construct an algorithm which does this approximately by recursively finding a tree tensor network of isometries which approximates U . This is done in an almost identical manner to that of section 6.1.

We start by associating the feature space \mathcal{H} with a lattice \mathcal{L} such that each site $s \in \mathcal{L}$ corresponds to one of the subspaces $\mathcal{H}_\phi \subset \mathcal{H}$. Denote the subspaces associated with site $s \in \mathcal{L}$ and set of sites $A \subset \mathcal{L}$ as \mathcal{H}_s and \mathcal{H}_A respectively. Now make a partition P of \mathcal{L} into disjoint blocks of neighbouring sites, and introduce a new lattice \mathcal{L}' in which each site s' corresponds to a block $B \in P$.

Just as before, we then wish to find a map between B and s' in terms of an isometry

$$w : \mathcal{H}_{s'} \rightarrow \mathcal{H}_B; \quad w^\dagger w = \text{id}, \quad (8.40)$$

and just as before, since we are most interested in preserving the dominating eigenvalues, the optimal choice is that which minimizes $\dim \mathcal{H}_{s'}$ while keeping

$$\frac{\text{tr } \rho - \text{tr } \rho_{s'}}{\text{tr } \rho} \leq \epsilon, \quad (8.41)$$

from which we can repeat the procedure of constructing reduced covariance matrices and finding optimal isometries.

Doing this recursively, we will eventually map the entire feature space to a single site, e.g. as

$$\mathbf{x}' = \mathbf{x}W = \begin{array}{c} \text{---} \\ | \\ \triangle \\ / \quad \backslash \\ \triangle \quad \triangle \\ / \quad \backslash \quad / \quad \backslash \\ \triangle \quad \triangle \quad \triangle \quad \triangle \\ | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \end{array} . \quad (8.50)$$

To see that this W has approximations of the eigenvectors of ρ as columns, consider first the case of $\epsilon = 0$. Then each w truncates only the null-space of the corresponding reduced covariance matrix and hence we get an exact diagonalization

$$\rho = W\Lambda W^\dagger = \begin{array}{c} \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \\ | \quad | \quad | \quad | \\ \triangle \quad \triangle \quad \triangle \quad \triangle \\ / \quad \backslash \quad / \quad \backslash \\ \triangle \quad \triangle \\ | \\ \triangle \\ | \\ \bullet \\ | \\ \triangle \\ / \quad \backslash \\ \triangle \quad \triangle \\ / \quad \backslash \quad / \quad \backslash \\ \triangle \quad \triangle \quad \triangle \quad \triangle \\ | \quad | \quad | \quad | \\ \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \end{array} . \quad (8.51)$$

For $\epsilon > 0$, this relation is of course no longer exact, but it is reasonable to assume that it gives a good approximation for sufficiently small ϵ .

8.4.2 Projection error

To give an estimation of the total truncation error, we will now derive limits on $\text{tr}[W^\dagger \rho W]$. This is simplest if we make a slight modification of the algorithm above. To this end we introduce an ordering of the blocks at each coarse-graining step, letting $\mathcal{L} = B_1 \cup B_2 \cup \dots \cup B_n$, and define the partially coarse-grained matrices

$$\rho_i \equiv \left(\bigotimes_{j \leq i} w_j^\dagger \right) \rho \left(\bigotimes_{k \leq i} w_k \right), \quad (8.52)$$

where w_i is the isometry acting on block B_i . We can then let the learning procedure for finding w_i be exactly as above, but with ρ everywhere replaced by ρ_{i-1} . Note that this leads to $\rho_0 = \rho$ and $\rho_i = \rho_{s'}$, giving the condition (8.41) as

$$\text{tr} \rho_i \geq (1 - \epsilon) \text{tr} \rho_{i-1}, \quad (8.53)$$

immediately giving a limit of the trace of the coarse-grained covariance matrix ρ' as

$$\text{tr} \rho' = \text{tr} \rho_n \geq (1 - \epsilon)^n \text{tr} \rho. \quad (8.54)$$

Continuing this recursively, we see that the truncation error of the fully coarse-grained system becomes as simple as

$$\mathcal{E} \equiv 1 - \frac{\text{tr}[W^\dagger \rho W]}{\text{tr} \rho} \leq 1 - (1 - \epsilon)^{N_{\text{iso}}} = N_{\text{iso}} \epsilon + \mathcal{O}(\epsilon^2) \quad (8.55)$$

where N_{iso} is the total number of isometries. Inverting this, we see that to guarantee a given total error \mathcal{E} we need to select

$$\epsilon \leq 1 - (1 - \mathcal{E})^{1/N_{\text{iso}}}. \quad (8.56)$$

Now consider the case of having feature vectors with M sites, where $\log_a M \in \mathbb{N}$ for some integer a . We can then construct a recursive coarse-graining scheme where each block contains a sites, giving the number of isometries at level l as M/a^l , and hence

$$N_{\text{iso}} = \sum_{l=1}^{\log_a M} \frac{M}{a^l} = \frac{M}{a} \frac{1 - a^{-\log_a M}}{1 - a^{-1}} = \frac{M - 1}{a - 1}. \quad (8.57)$$

This truncation error estimation highlights the sensitivity in ϵ which was empirically found by [18]; since M is for an image at least on the order of 10^3 , while a needs to be chosen to a small integer, N_{iso} will often be on the same order of magnitude as M . By (8.55) this gives a total truncation error which is very sensitive to the choice of ϵ .

9

Quantum entanglement in machine learning

There has recently been much interest [24, 63–68] in comparing and evaluating different representational structures for the functions used in ML. A particular concern is that of expressivity, which loosely may be seen as evaluating the efficiency (typically per parameter) of a given neural network in representing a wide class of functions. Although popular as a research topic, there is as of now no consensus on how to quantify the expressivity of a network, and it is also possible that different situations may call for different measures of this quantity.

The properties to demand from such a measure are:

1. It should have a clear interpretation from the point of view of measure theory or statistics, so as to allow informed reasoning on what to expect from a model with a given expressivity.
2. It should be contractible, and ideally inferable directly from the structure of a model, in order to be useful as a tool to instruct construction of new models.

If we consider generative models where the function to represent is simply the probability distribution of the data itself, an obvious starting point is to simply consider which correlations can be modelled between the input variables; e.g. given a partition of the variables into X and Y , one might wish to give bounds on the mutual information $I(X : Y)$.

Often, however, the probabilistic interpretation is lacking. We might instead wish to consider the *separation rank*[63] between sets of variables. For a function $f(X, Y)$ this is given by the minimum number of terms r in the expansion

$$f(X, Y) = \sum_{i=1}^r a_i \psi_i(X) \varphi_i(Y), \quad (9.1)$$

where $a_i \in \mathbb{C}$ and ψ_i and φ_i may be chosen as (orthonormal) basis functions. Of course, this assumes that such an expansion is possible at all, but this should be the case so long as f is square-integrable.

Although the separation rank may be a little unclear interpretationally it is – as we shall see – easily connected to the Schmidt number, or equivalently the zeroth quantum Rényi entropy. If we further can express the function on the form of a nontrivial TN, as with all the models considered in the previous chapter, the quantum max-flow/min-cut theorem makes it a simple matter to infer upper, and in some cases also lower, limits on the separation rank directly from the graph of the TN.

9.1 Rényi entropy of TNL models

To make things a bit more formal, we start by considering the setting in section 8.1 of having multivariate models $f(x)$ which can be expressed on the form

$$f(x) = W \cdot \Phi(x), \quad \Phi(x) = \phi(x_1) \otimes \phi(x_2) \otimes \dots \otimes \phi(x_N), \quad (9.2)$$

either due to f being square-integrable or to some explicit local feature map. Here $\phi \in \mathcal{H}_\phi$, $\Phi \in \mathcal{H} \equiv \mathcal{H}_\phi^{\otimes N}$ and $W \in \text{hom}(\mathcal{H}, \mathcal{H}_{\text{out}})$ with $\mathcal{H}_\phi, \mathcal{H}$ and \mathcal{H}_{out} being (ideally finite-dimensional) Hilbert spaces. For the sake of interpretability it is convenient to also demand that $\{\Phi_{i_1, i_2, \dots, i_N}\}_{\{i_k\}}$ forms an orthonormal basis over \mathcal{H} .

As noted previously, f is now on a form which is similar to that of an unnormalized quantum many-body wave function. Restricting to $\mathcal{H}_{\text{out}} = \mathbb{C}$ or even $\mathcal{H}_{\text{out}} = \mathbb{R}$, the situation is (more or less) identical and, encouraged by this similarity, we may form a normalized density matrix for f through

$$\rho \equiv \frac{W^\dagger W}{\text{tr}[W^\dagger W]} = \frac{W^\dagger \otimes W}{\text{tr}[W^\dagger \otimes W]}. \quad (9.3)$$

Note that both of the above expressions are also available for $\dim \mathcal{H}_{\text{out}} > 1$, but they are then inequivalent since we with $W^\dagger W$ imply taking the inner product over \mathcal{H}_{out} .

Naively continuing onward, we are now ready to consider the entanglement entropies of ρ . First of all, when $\rho \propto W^\dagger \otimes W$ as is always the case for $\dim \mathcal{H}_{\text{out}} = 1$, ρ is a pure state, immediately giving $S_\alpha(\rho) = 0$.

To look at correlations we partition the inputs into two sets A and $B = A^c$ and define

$$\begin{cases} \Phi_A(x) \equiv \bigotimes_{i \in A} \phi(x_i) & \in \mathcal{H}_A \\ \Phi_B(x) \equiv \bigotimes_{i \in B} \phi(x_i) & \in \mathcal{H}_B \end{cases} \quad (9.4)$$

so that $\Phi(x) = \Phi_A(x) \otimes \Phi_B(x)$ and $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$.

Making a Schmidt decomposition of W with respect to the (A, B) partition gives

$$W = \Lambda(U_A \otimes U_B), \quad (9.5)$$

where $\Lambda_{ij} = \sum_{k=1}^r \lambda_k \delta^k_{ij}$, with λ_k being the singular values and r the rank of the decomposition, and U_A and U_B are unitary operators in $\text{hom}(\mathcal{H}_A, \mathbb{C}^r)$ and $\text{hom}(\mathcal{H}_B, \mathbb{C}^r)$ respectively. Noticing that this puts $f(x) = W \cdot \Phi(x)$ on the form of equation (9.1), this shows that the Schmidt number r is the separation rank of $f(x)$ w.r.t. (A, B) .

Continuing toward the full quantum Rényi entropy, we introduce the reduced density matrix

$$\rho_A \equiv \text{tr}_B \rho = \frac{U_A^\dagger \Lambda_A U_A}{\text{tr} \Lambda_A}, \quad (9.6)$$

where $\Lambda_A \equiv \text{diag}\{|\lambda_1|^2, |\lambda_2|^2, \dots\}$. This gives

$$S_\alpha(\rho_A) = \frac{1}{1-\alpha} \log \text{tr}(\rho_A^\alpha) = \frac{1}{1-\alpha} \log \frac{\text{tr} \Lambda_A^\alpha}{(\text{tr} \Lambda_A)^\alpha} = \frac{1}{1-\alpha} \log \frac{\sum_i |\lambda_i|^{2\alpha}}{(\sum_i |\lambda_i|^2)^\alpha}. \quad (9.7)$$

It is also of interest to repeat the results that $r = 1$ gives $S_\alpha(\rho_A) = 0$, that $S_0(\rho_A) = \log r$ and that $S_\alpha(\rho_A) = S_\alpha(\rho_B)$.

This is of course identical to what we found already in the chapter on quantum information, although the λ_i can no longer be interpreted as probability amplitudes. Rather, the interpretation of $S_\alpha(\rho_A)$ is simply as a measure of how far $f(x)$ is from being separable with respect to the partition (A, B) . Here the parameter α decides how the singular values are weighted, with $\alpha = 0$ disregarding the sizes of the singular values completely and $\alpha \gg 1$ emphasizing the larger on expense of the smaller.

In the case of $f(x)$ being a probability distribution, or a probability amplitude (i.e. $p(x) = |f(x)|^2$), separability implies statistical independence and we may hence in these cases think of $S_\alpha(\rho_A)$ as a measure of correlation between A and B . As noted by [69], there is some (superficial) similarity between the reduced quantum Rényi entropy considered here and the Shannon mutual information $I(A; B)$, although it is unclear whether this similarity amounts to anything more than both being measures of correlation.

9.2 Boltzmann machines

A straightforward application of quantum entropy for analysis of machine learning architectures is on Boltzmann machines, as considered by [70, 71]. For consistency with these two articles we will consider the RBM to model a probability amplitude $\psi(v)$ over some visible variables $v_i \in \{0, 1\}$, $i = 1, \dots, N$, such that $p(v) \equiv |\psi(v)|^2$ and $\sum_v p(v) = |\psi|^2 = 1$. This amplitude is then given by

$$\psi(v) = \sum_{\{h_i\}} \frac{1}{Z} e^{-E(v, h)}, \quad E(x) = -\langle Wx, x \rangle - \langle x, a \rangle, \quad (9.8)$$

where $x = (v, h)$ and W is upper triangular such that $W_{ij} = 0$ if $i \geq j$.

9.2.1 Mapping to tensor networks

Since $v \in \{0, 1\}^N$, we have that $\psi \in (\mathbb{C}^2)^{\otimes N}$, i.e. ψ is a vector in a tensor product Hilbert space. This makes it natural to try and express ψ on the form of a TN in order to find limits on its entanglement entropy. In fact, this turns out to be possible by making slight modifications on the graph of the BM [71]:

Keep all nodes in the network, but add a dangling edge to each visible node v_i and let the tensor corresponding to node x_i be

$$\Lambda^{(i)} \equiv \text{Diag}\{1, e^{a_i}\}. \quad (9.9)$$

For each non-zero W_{ij} add a node on the edge connecting x_i and x_j , and let the corresponding tensor be

$$M^{(ij)} \equiv \begin{pmatrix} 1 & 1 \\ 1 & e^{-W_{ij}} \end{pmatrix}. \quad (9.10)$$

It may then be seen that ψ is exactly given by

$$\psi(v_1, \dots, v_N) = \left[\left(\prod_{i,j} M^{(ij)} \right) \left(\prod_k \Lambda^{(k)} \right) \right]_{v_1, \dots, v_N}, \quad (9.11)$$

where inner products are taken according to the graph and we have left out the scalar factor of $1/Z$ for convenience. For a simple example, see figure 9.1.

If we are interested in using the QMF / QMC theorems of section 5.2 for inferring limits on entanglement from the TN, we may safely ignore the $M^{(ij)}$ tensors, since they are

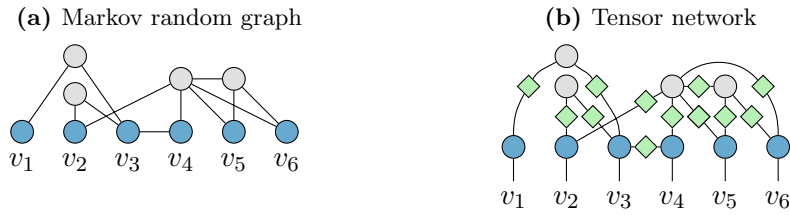


Figure 9.1: The Boltzmann machine represented by the Markov random graph in (a) models the same function as the tensor network in (b). The light grey dots represent hidden units, the dark blue dots are visible units and the green diamonds are the matrices $M^{(ij)}$. Hidden and visible units h_i and v_j in the random graph are replaced by tensors $\Lambda_h^{(i)}$ and $\Lambda_v^{(j)}$ respectively in the tensor network.

full-rank matrices for all $W_{ij} \neq 0$ and hence does not affect the connectivity of the graph. We are hence left with a TN on the same graph as the BM, but with dangling edges on the visible nodes. Letting $\text{QMC}_r(A)$ and $\text{QMF}_r(A)$ denote the restricted quantum min cut and quantum max flow implied by restricting the Λ tensors to being diagonal and interpreting the graph as a flow network from A to A^c , with $A \cup A^c$ being the set of inputs to the network.

By the restricted quantum max flow/min cut theorem,

$$S_\alpha(A) \leq \log \text{QMC}_r(A) = \log 2^{\text{MC}_r(A)} = \text{MC}_r(A) \log 2, \quad (9.12)$$

where we take $\text{MC}_r(A)$ to be the unweighted combined edge-vertex min cut separating A from A^c .

9.2.2 Boltzmann machines as MPS

It is an interesting problem to try and construct the optimal MPS for a given Boltzmann machine. A constructive approach to this problem is to start from the TN above and perform contractions and merge edges in an appropriate order until the TN is on the structure of an MPS.

We may as a first step contract all the $M^{(ij)}$ arbitrarily, since this does not affect the connectivity of the graph. Had we started from the BM in figure 9.1, this would leave us with the TN

$$\psi = \text{[Diagram of TN with contracted matrices]} \quad (9.13)$$

Next we find successive min cuts C_i , $i = 1, \dots, N - 1$ such that C_i is the smallest cut separating the visible units $\{v_1, \dots, v_i\}$ from $\{v_{i+1}, \dots, v_N\}$. These cuts partition the TN into N TNs which are then separately contracted into the multimode tensors $A^{(i)}$, or in our example

$$\text{[Diagram of TN with cuts C1-C5]} = A^{(1)} A^{(2)} A^{(3)} A^{(4)} A^{(5)} A^{(6)}, \quad (9.14)$$

where

$$\begin{aligned}
 A^{(1)} &= \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array}, & A^{(2)} &= \begin{array}{c} \bullet \\ \text{---} \\ \text{---} \\ \text{---} \end{array}, & A^{(3)} &= \begin{array}{c} \bullet \\ \text{---} \\ \bullet \\ \text{---} \\ \bullet \end{array}, \\
 A^{(4)} &= \begin{array}{c} \text{---} \\ \bullet \\ \text{---} \end{array}, & A^{(5)} &= \begin{array}{c} \bullet \\ \text{---} \\ \bullet \\ \text{---} \\ \bullet \end{array}, & A^{(6)} &= \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array}.
 \end{aligned} \tag{9.15}$$

The tensor $A^{(i)}$ now has one edge to v_i but $|C_{i-1}|$ and $|C_i|$ edges to the neighbouring tensors $A^{(i-1)}$ and $A^{(i+1)}$ respectively. The last step is to merge all parallel edges in the TN. Since the unmerged edges all correspond to an inner product over 2-dimensional Hilbert spaces, merging all $|C_i|$ edges between tensors $A^{(i)}$ and $A^{(i+1)}$ gives an edge of dimension $2^{|C_i|}$. We are hence left with an MPS where the i th internal bond has dimension $2^{|C_i|}$. Our example BM hence becomes

$$\psi = \begin{array}{c} \square \\ | \\ v_1 \end{array} \begin{array}{c} \square \\ | \\ v_2 \end{array} \begin{array}{c} \square \\ | \\ v_3 \end{array} \begin{array}{c} \square \\ | \\ v_4 \end{array} \begin{array}{c} \square \\ | \\ v_5 \end{array} \begin{array}{c} \square \\ | \\ v_6 \end{array}, \tag{9.16}$$

where the thin lines have dimension 2 and the thick lines dimension 4.

This contraction procedure creates an MPS which is smaller than or of equal size to what can be reached with the similar procedure suggested for the special case of RBMs in [71]. The same article do however derive a more intricate algorithm, which they claim to find the optimal MPS for a given RBM and which does outperform the above procedure.

9.2.3 Area law of entanglement for local RBMs

Following [70] we will now show that the upper limit of entanglement entropy follows an area law for local RBMs, which we will now define. Consider first an RBM on the form above, where both the visible and the hidden units are placed on a d -dimensional lattice \mathcal{L} . Call these units $v_r \in \{0,1\}$ and $h_r \in \{0,1\}$ respectively, where $\mathbf{r} = (r_1, \dots, r_d) \in \mathcal{L}$. A *local RBM* of range $R \geq 0$ is then an RBM where the weights $W_{\mathbf{r},\mathbf{r}'}$ are only non-zero for $|\mathbf{r} - \mathbf{r}'| \leq R$. We are interested in the entanglement $S_\alpha(A)$ between the visible units of a volume $A \subset \mathcal{L}$ and the visible units in A^c .

We know from the previous sections that in the TN representation of a BM, $S_\alpha(A) \leq \text{MC}_r(A) \log 2$, where $\text{MC}_r(A)$ is the unweighted combined min cut separating A from A^c . In an R -range RBM, only hidden units within a distance of R from the surface ∂A of A will have edges crossing the surface. Since removing a single node is equivalent to cutting any number of edges connected to the node, the combined minimal cut is the same as the number of vertices within this distance from ∂A , giving immediately

$$\text{MC}(A) \leq 2RS(A), \tag{9.17}$$

where $\mathcal{S}(A)$ is the area of ∂A . This then leads to

$$S_\alpha(A) \leq 2 \log 2 RS(A), \tag{9.18}$$

showing that the entanglement entropy of a subsystem in a local RBMs is indeed bounded by its area. This bond is identical to what was found with a different method by [70].

9.2.4 Area law of entanglement for local DBMs

We define a *local deep Boltzmann machine* of range R to be a DBM built by stacking local RBMs of range R on top of each other; that is, we place all units v_r and h_r^l , $l = 1, \dots, L$ on a d -dimensional lattice \mathcal{L} and let $W_{\mathbf{r},\mathbf{r}'}^l = 0$ if $|\mathbf{r} - \mathbf{r}'| > R$.

Now it turns out that, to disconnect A from A^c , you need to remove from every odd hidden layer, the same number of nodes as we found for the single-layer RBM. This gives our upper bound for the quantum Rényi entropy in the local DBM as

$$S_\alpha(A) \leq 2 \log 2 \left\lceil \frac{L}{2} \right\rceil R \mathcal{S}(A), \quad (9.19)$$

which, interestingly, is also an area law, confirming what was found with a different method by the recent paper [72].

Note also that the entropy is linear both in L and R , while the number of parameters scales like $\sim L$ and $\sim R^D$. It is hence cheaper to increase entanglement by adding layers than by increasing range, giving an indication that DBMs possess depth efficiency.

9.2.5 Volume law of entanglement for long range BMs and RBMs

Clearly, if R is selected large enough that $R\mathcal{S}(A) \geq \text{vol}(A)$, the minimal combined cut is going to be that which removes all vertices corresponding to visible variables in A , and we will have instead

$$S_\alpha(A) \leq \log 2 \text{ vol}(A). \quad (9.20)$$

Note that this limit is just the maximum possible entanglement for any system of binary variables.

9.3 Convolutional arithmetic circuits

The introduction of quantum entropy as a measure of expressivity in ML was originally proposed in the context of convolutional arithmetic circuits by Levine et al[24]. We will in this section make a brief review of their results and generalize them somewhat using the machinery developed in the sections on quantum max flow/min cut (see 5.2.2).

We will however, for convenience, change the notation compared to section 5.2 somewhat. For an nCAC ψ with graph G , quantum capacity function c , and with A as a subset of the dangling edges of G (or equivalently, inputs to ψ), we will let $\text{QMC}(A, \psi)$ be what we previously denoted by $\text{QMC}(G, c)$ with G interpreted as a flow network from A to A^c . Similarly we will replace $\text{QMF}(G, c)$ with $\text{QMF}(A, \psi)$ and the restricted versions $\text{QMC}(G, c; U)$ and $\text{QMF}(G, c; \mathcal{J})$ with $\text{QMC}_r(A, \psi)$ and $\text{QMF}_r(A, \psi)$. Here U is the set of vertices for which tensors are restricted to be diagonal while $\mathcal{J}(\psi)$ is the allowed configuration space for the tensors. For CACs and nCACs, it is evident that U is the set of vertices with Kronecker delta tensors. Let \mathcal{J}_U be the largest configuration space for which U are Kronecker deltas. Then, for an nCAC we have $\mathcal{J}(\psi) = \mathcal{J}_U$, while for a CAC, $\mathcal{J}(\psi)$ is a proper subset of \mathcal{J}_U .

It should be noted that much of the results in this section are developed for nCACs¹. They are still somewhat relevant also for CACs since they may be viewed as being restrictions of nCACs. An upper limit on quantum entanglement derived for an nCAC will hence hold also for a regular CAC, although the same can of course not be said about the lower limit.

9.3.1 Depth efficiency

The use of quantum entropy considered by [24] is mainly to explain the depth efficiency of nCACs by comparing two non-overlapping nCACs acting on N features where $\log_2 N \in \mathbb{N}$;

¹Near-/non-convolutional arithmetic circuits, see section 8.3.2.

a deep one with $S = K = 2$ and a shallow one with $S = K = N$. For convenience they specialize to networks which give scalar outputs; i.e. such that $f(x) = W\Phi(x) \in \mathbb{R}$ (or some other field).

The analysis of the shallow model is very simple; having $S = K = N$ reduces the model to having a single layer, with the TN representation of W being on the form

$$W = \begin{array}{c} \textcircled{g} \\ \triangle \\ \textcircled{g} \quad \textcircled{g} \quad \dots \quad \textcircled{g} \end{array} \quad (9.21)$$

Let c be the number of channels after the first matrix product, or equivalently the quantum capacity of the edges connected to the Kronecker delta. Since δ_r has all matricization ranks equal to r , this gives the upper limit on any entanglement entropy in the shallow network as $S_\alpha \leq \log r$.

For the deep model Levine et al arrive at the following:

Lemma 9.1. Consider an nCAC ψ^p with kernel size and stride $K = S = 2$, such that all bond dimensions are integer powers of p and consider a subset A of the inputs to the network. Then $S_0(A) = \log \text{QMC}_r(A, \psi)$ for all assignments of the tensors in ψ^p , except for a set with Lebesgue measure zero.

Proof. See [24]. □

This leads to the following statement for general bond dimensions.

Theorem 9.1. Consider an nCAC ψ with kernel size and stride $K = S = 2$, graph $G = (V, E)$ and bond dimensions $c : E \rightarrow \mathbb{N}$ and define ψ^p with $p \in \mathbb{N}$, to be an nCAC on the same graph but with bond dimensions $c_p(e) \equiv p^{\lfloor \log_p c(e) \rfloor}$. Consider a subset A of the inputs to the network. Then $S_0(A) \geq \log \text{QMC}(A, \psi^p)$ for all assignments of the tensors in ψ , except for a set with Lebesgue measure zero.

Proof. See [24]. □

It is clear that, with appropriately chosen region A , this can give much larger values for $S_0(A)$ in a deep nCAC than is possible with a shallow one. However, do note that the tree structure of the graph also makes it possible to find arbitrarily large input regions which may be disconnected from their complement by cutting a single bond.

By utilizing the toolbox on max flow/min cut developed in section 5.2.2 we may in fact arrive at versions of the above statements which encompass all possible choices of S and K with $S = K$.

Lemma 9.2. Consider an nCAC ψ^p with kernel size and stride $K = S \leq N$, such that all bond dimensions are integer powers of p and consider a subset A of the inputs to the network. Then $S_0(A) = \log \text{QMC}_r(A, \psi)$ for all assignments of the tensors in ψ^p , except for a set with Lebesgue measure zero.

Proof. It is an immediate consequence of theorem 5.2 that $\text{QMF}_r(A, \psi^p) = \text{QMC}_r(A, \psi^p)$. The statement of the theorem is then arrived at by employing proposition 5.3. □

Theorem 9.2. Consider an nCAC ψ with kernel size and stride $K = S \leq N$, graph $G = (V, E)$ and bond dimensions $c : E \rightarrow \mathbb{N}$ and define ψ^p with $p \in \mathbb{N}$, to be an nCAC on

the same graph but with bond dimensions $c_p(e) \equiv p^{\lfloor \log_p c(e) \rfloor}$. Consider a subset A of the inputs to the network. Then $S_0(A) \geq \log \text{QMC}_r(A, \psi^p)$ for all assignments of the tensors in ψ , except for a set with Lebesgue measure zero.

Proof. The set of tensors realizable by ψ^p is clearly a subset of those realizable by ψ , giving $\text{QMF}_r(A, \psi) \geq \text{QMF}_r(A, \psi^p)$. By the previous lemma, $\text{QMF}_r(A, \psi^p) = \text{QMC}_r(A, \psi^p)$, which leads us to $\text{QMF}_r(A, \psi) \geq \text{QMC}_r(A, \psi^p)$. By proposition 5.3, $S_0(A) = \text{QMF}_r(A, \psi)$ almost everywhere, finishing the proof. \square

It is clear that if A is a connected region in the input space, the minimal cut will consist of at most $\mathcal{S}(A)$ vertices or edges at each level. Hence for the 1D non-overlapping nCAC (as well as the CAC) with quantum capacity everywhere set to $c(e) = r$, we have

$$S_\alpha(A) \leq \log r^{2 \log_K |A|} = \frac{2 \log r}{\log K} \log |A|. \quad (9.22)$$

Since depth increases as K decreases, this does indicate that deep networks are able to model more intricate connections than shallow ones. To make this even more clear, we note that the number of parameters in an nCAC with $N = K^{L-1}$ inputs and all quantum capacities equal to r is

$$(K^{L-1} + K^{L-2} + \dots + 1)r^2 = (K^L - 1)r^2 = \mathcal{O}(N^{L/L-1}) \xrightarrow{L \gg 1} \mathcal{O}(N). \quad (9.23)$$

However, to reach the same $S_\alpha(A)$ for $|A| = N/K$ with the shallow model, the quantum capacities of the $N + 1$ tensors would need to be

$$r' = r^{2(L-2)} = \frac{K^{2(L-1) \log_K r}}{r^2} = \frac{N^{2 \log_K r}}{r^2} \quad (9.24)$$

for a total of

$$(N + 1) \frac{N^{4 \log_K r}}{r^4} = \mathcal{O}(N^{4 \log_K r + 1}) \quad (9.25)$$

parameters.

For large N , this makes the deep model more parameter efficient (for certain choices of A) when

$$r > K^{\frac{1}{4}}. \quad (9.26)$$

However, once again it should be noted that the tree structure of the model reduces the minimal cut to one edge for arbitrarily sized special choices of A .

10

Discussion

We have in this thesis sought to answer two separate, but interrelated questions: “*To what extent can algorithms from numerical quantum mechanics be employed for general machine learning?*” and “*To what extent can quantum information theory be used to explain and understand different models in ML?*”.

In order to answer the first question, a literature review of current QM-inspired methods in ML was conducted in chapter 8, with the central finding that most of the work being done in this direction employ the tensor networks introduced in chapter 5. The second question was discussed in chapter 9 where we follow e.g. [24, 69] in evaluating quantum entanglement as a measure of expressiveness in a neural network. This analysis also led to the discovery of a slight generalization of the quantum max flow/min cut theorem by Cui et al[25], which is introduced in section 5.2.

Since the subjects of tensor network learning, quantum max-flow/min-cut and entanglement as a measure of expressiveness, while relevant for each other, are quite separate, we have divided this chapter into three sections correspondingly.

10.1 Numerical quantum mechanics for machine learning

The idea to employ techniques from numerical quantum mechanics originates to large degree from the notion that the learning processes of deep neural networks are similar to a renormalization group flow [13]. It is hence no surprise that the numerical renormalization procedures from quantum many-body theory plays a large part in this endeavour. Particularly models which utilize tensor networks to create (multi-)linear functions in high dimensional vector spaces have become very popular; in addition to the models discussed in this thesis, several models have been proposed, for both supervised and unsupervised learning, most often building on tree tensor networks[23, 60] or MPS[61].

There is actually a quite interesting and somewhat plausible difference in interpretation between the models discussed in this thesis and those of standard DNN. In standard DNN’s every few layers contain a non-linearity which may be thought of as analogues to the local feature maps of TNL. Since weights are updated both before and after each feature map, the feature maps themselves change as the network learns, with the interpretation that it *learns* the relevant features of the data it is fed.

In contrast, the TNL algorithms are fed an impossibly large number of features and tasked with creating a linear model based on these. Since the linear model by necessity have a much smaller output dimension than input, its output will be a linear combination of a subset of the features. Hence we can think of the learning of these algorithms as *finding* the relevant features of data amongst a pre-existing set.

This also highlights one of the possible drawbacks to this method; while the features of DNNs are variable and may be adjusted continuously, the features of TNL are fixed and thus need to be chosen appropriately from the start. Because of this, it would be very

interesting to see what would be achievable by stacking several layers of linear models in tensor product spaces on top of each other, with non-linearities between each one. In particular the tree curtain model by [18] seems to be a good candidate to extend.

It is remarkable to note that, despite the recent hype around the MERA tensor network in the physics community, there has been very few reported attempts to employ it for ML. The one example we can find is [21], where a quite thinned down version of the MERA is used in a similar manner to the unsupervised coarse graining of [18]. A probable reason for the relative lack of MERA-based algorithms is that, in contrast to TTNs, a product state in one layer is not mapped into a product state in the next, which makes calculating the inner product between a MERA and a product state a computationally much more expensive process. It is nevertheless a very interesting direction to consider for future research, as the issue of entanglement being highly dependent on how the geometry of the partition relates to the graph which is prevalent in TTNs is avoided in the MERA.

Another possible direction of future research is to take a more general look at which graphs make for TNs which are good for ML. In particular, graphs with loops seem to be generally avoided, which is sensible since loops increase the complexity of contraction. However, loops also have an essential role in distributing entanglement and our intuition is that there may be methods to introduce them without the network becoming intractable. One idea in this direction would be to take inspiration of the TN representation of Boltzmann machines and generalize to arbitrary TNs on similar graphs, but keeping tensors diagonal in the BM representation diagonal also in the generalized case.

10.2 Entanglement analysis of ML architectures

We have in this thesis studied the use of quantum entanglement entropy as a measure of the expressivity of a neural network and found that it is indeed sensible when viewed as measuring the distance from multiplicative separability with regards to partitions of the inputs. It is, however, slightly difficult to interpret outside of the context of QM, giving reason for some concern for its usefulness. Fortunately, much the analysis done in chapter 9 is highly relevant also for the condensed matter community, where large efforts (see [73] for an excellent review) are now being made to use machine learning techniques to simulate highly correlated quantum systems. A major concern is then how to choose the representation of the wave function in order to be able to reach sufficiently entangled states, and in this context, the entanglement properties of particularly RBMs and DBMs[70–72, 74, 75], but also other architectures[76] have been well studied.

Out of what is studied in chapter 9, particularly the area-law scaling of entanglement in local RBMs[70] and DBMs[72], logarithmic breaking of the area law in non-overlapping nCACs[24], and volume law in fully connected RBMs and DBMs become very relevant (volume law scaling is shown also for overlapping nCACs in [64, 77]). In this context, we produce no new results, but do however with the help of our developments on QMF/QMC provide alternate, and in our opinion often simpler, proofs than the original sources. Hopefully, this together with the QMF / QMC results which we will bring up next, may lead to further insights in the future.

We also wish to note that almost all of chapter 9 deals exclusively with the zeroth Rényi entropy of different partitions, or equivalently the multi-rank of the tensor represented by the TN. As measure of distance from separability, this is rather crude – if an expansion

$$f(x,y) = \sum_{i=1}^r a_i \psi_i(x) \varphi_i(y) \quad (10.1)$$

has nearly all of its weight in the first term, we would ideally consider f to be very close to being separable. However, $S_0(X) = \log r$, independent of the assignments of a_i . On the one hand, this very independence is what makes it possible to estimate the zeroth Rényi entropy without knowledge of the actual tensors, but on the other, the information one can get from merely knowing S_0 becomes quite limited. As such, it would be an interesting endeavour to estimate other orders of the Rényi entropy from TNs, particularly when viewing the tensors as randomly distributed. This has to some extent already been studied for TNs with unitary tensors in the AdS/CFT community – see e.g. [43] who show that the problem of finding the expected value of the second Rényi entropy in certain cases can be reduced to that of minimizing the free energy of a Boltzmann machine on the same graph. Nonetheless, it would be interesting to see what is achievable by taking a more holistic view.

10.3 Quantum max-flow/min-cut

It is long known that the quantum entropy in TNs is related to minimal cuts in the corresponding graphs, a fact which was elaborated on in the quantum max flow/min cut theorem in [25], making the entanglement analysis of a model a simple matter once a map to an equivalent TN can be found. We have studied the TNs of Boltzmann machines as found by [71] and CACs as found by [24]. Common to these two is a prevalence of diagonal tensors, which have the property that all their matricization ranks are equal, making the cutting of an edge connected to the tensor essentially equivalent to removing the vertex altogether. This was elaborated on in section 5.2.2 where a stronger quantum max flow/min cut result was achieved for TNs with diagonal tensors, using the concept of combined cuts introduced in section 4.4.

There is in fact a sense in which the more natural cut for a quantum max flow analysis is a weighted vertex cut, where each vertex is given a variable weight chosen from its tensor's multi-rank according to which matricization is implied by the cut. Our hypothesis is that $\text{QMF}_r = \text{QMC}_r$ would hold to a similar extent also in this case, while allowing a more fine-grained view of the entanglement features of a restricted TN. There is a cost to this approach, however, in that it both requires more information of the tensors and that the calculation of a minimal cut may become much more computationally expensive.

Another problem which we encountered in doing QMF / QMC analysis on CACs is that it is presently not known whether some version of $\text{QMF} = \text{QMC}$ holds for translationally invariant networks, or more generally, networks where some tensors are restricted to be equal to each other. It would be very interesting to see some analysis done on this problem, since the case of having equal tensors is very common in the literature. In particular, the scale invariance which makes MERA relevant from a conformal field theory (and hence also quantum gravity) perspective hinges on the equality of tensors at different scales.

Bibliography

- [1] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* **5**(4), pp. 115–133 (1943).
- [2] D. O. Hebb. *The Organization of Behaviour*. John Wiley and Sons, New York (1949).
- [3] B. Farley and W. Clark. Simulation of self-organizing systems by digital computer. *Transactions of the IRE Professional Group on Information Theory* **4**(4), pp. 76–84 (1954).
- [4] Y. Huang et al. GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism (2018). [arXiv:1811.06965](https://arxiv.org/abs/1811.06965).
- [5] G. Saon et al. English Conversational Telephone Speech Recognition by Humans and Machines (2017). [arXiv:1703.02136](https://arxiv.org/abs/1703.02136).
- [6] G. Lample et al. Phrase-Based & Neural Unsupervised Machine Translation (2018). [arXiv:1804.07755](https://arxiv.org/abs/1804.07755).
- [7] Papers With Code : the latest in machine learning. <https://paperswithcode.com/>. (visited 2019-05-14).
- [8] G. E. Hinton, S. Osindero and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation* **18**(7), pp. 1527–1554 (2006).
- [9] H. W. Lin, M. Tegmark and D. Rolnick. Why Does Deep and Cheap Learning Work So Well? *Journal of Statistical Physics* **168**(6), pp. 1223–1247 (2017).
- [10] D. J. Schwab and P. Mehta. Comment on "Why does deep and cheap learning work so well?" (2016). [arXiv:1609.03541](https://arxiv.org/abs/1609.03541).
- [11] S. Iso, S. Shiba and S. Yokoo. Scale-invariant feature extraction of neural network and renormalization group flow. *Physical Review E* **97**(5), pp. 1–32 (2018). [arXiv:1801.07172v1](https://arxiv.org/abs/1801.07172v1).
- [12] M. Koch-Janusz and Z. Ringel. Mutual information, neural networks and the renormalization group. *Nature Physics* **14**(6), pp. 578–582 (2018).
- [13] P. Mehta and D. J. Schwab. An exact mapping between the Variational Renormalization Group and Deep Learning (2014). [arXiv:1410.3831](https://arxiv.org/abs/1410.3831).
- [14] S. R. White. Density matrix formulation for quantum renormalization groups. *Physical Review Letters* **69**(19), pp. 2863–2866 (1992).
- [15] S. R. White. Density-matrix algorithms for quantum renormalization groups. *Physical Review B* **48**(14), p. 10345 (1993).

- [16] G. Vidal. Entanglement renormalization. *Physical review letters* **99**(22), p. 220405 (2007). [arXiv:cond-mat/0512165](#).
- [17] E. Stoudenmire and D. J. Schwab. Supervised learning with tensor networks. *Advances in Neural Information Processing Systems*, pp. 4799–4807 (2016). [arXiv:1605.05775](#).
- [18] E. M. Stoudenmire. Learning relevant features of data with multi-scale tensor networks. *Quantum Science and Technology* **3**(3), pp. 1–12 (2018). [arXiv:1801.00315v1](#).
- [19] Y. Liu, X. Zhang, M. Lewenstein and S.-J. Ran. Entanglement-guided architectures of machine learning by quantum tensor network (2018). [arXiv:1803.09111](#).
- [20] Z.-Z. Sun et al. Generative Tensor Network Classification Model for Supervised Machine Learning (2019). [arXiv:1903.10742](#).
- [21] G. Evenbly. Number-State Preserving Tensor Networks as Classifiers for Supervised Learning (2019). [arXiv:1905.06352](#).
- [22] S. Efthymiou, J. Hidary and S. Leichenauer. TensorNetwork for Machine Learning (2019). [arXiv:1906.06329](#).
- [23] S. Cheng, L. Wang, T. Xiang and P. Zhang. Tree tensor networks for generative modeling. *Physical Review B* **99**(15), p. 155131 (2019). [arXiv:1901.02217](#).
- [24] Y. Levine, D. Yakira, N. Cohen and A. Shashua. Deep Learning and Quantum Entanglement: Fundamental Connections with Implications to Network Design. *International Conference on Learning Representations (ICLR)* (2018). [arXiv:1704.01552](#).
- [25] S. X. Cui et al. Quantum Max-flow/Min-cut. *Journal of Mathematical Physics* **57**(6), pp. 1–28 (2016). [arXiv:1508.04644](#).
- [26] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal* **27**(3), pp. 379–423 (1948).
- [27] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons. ISBN 9780471241959 (2005).
- [28] M. Mézard and A. Montanari. *Information, physics, and computation*. Oxford graduate texts. Oxford University Press, Oxford; New York. ISBN 0-19-857083-X (2009).
- [29] E. Witten. *A Mini-Introduction To Information Theory* (2018). [arXiv:1805.11965](#).
- [30] M. Wilde. *Quantum information theory*. Cambridge University Press, second edn. ISBN 978-1-107-17616-4 (2017).
- [31] D. K. Fadeev. Zum Begriff der Entropie einer endlichen Wahrscheinlichkeitsschemas. *Arbeiten zur Informationstheorie I. Deutscher Verlag der Wissenschaften* pp. 85–90 (1957).
- [32] A. Renyi. On measures of entropy and information. *Proc. Fourth Berkeley Symp. on Math. Statist. and Prob.* (1961).
- [33] P. Erdos. On the distribution function of additive functions. *Annals of Mathematics* **47**(1), pp. 1–20 (1946).

-
- [34] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics* **22**(1), pp. 79–86 (1951).
- [35] B. B. Machta, R. Chachra, M. K. Transtrum and J. P. Sethna. Parameter space compression underlies emergent theories and predictive models. *Science (New York, N.Y.)* **342**(6158), pp. 604–7 (2013).
- [36] D. Xu and D. Erdogmus. Renyi’s Entropy, Divergence and Their Nonparametric Estimators. J. C. Principe (ed.), *Information Theoretic Learning, Information Science and Statistics*, chap. 2, pp. 47–102. Springer New York, New York, NY. ISBN 978-1-4419-1570-2 (2010).
- [37] I. Goodfellow, Y. Bengio and A. Courville. *Deep Learning*. MIT Press (2016).
- [38] J. C. Bridgeman and C. T. Chubb. Hand-waving and interpretive dance: An introductory course on tensor networks. *Journal of Physics A: Mathematical and Theoretical* **50**(22) (2017). [arXiv:1603.03039](https://arxiv.org/abs/1603.03039).
- [39] L. Dixon and A. Kissinger. Open Graphs and Monoidal Theories. *Mathematical Structures in Computer Science* **23**(2), pp. 308–359 (2010). [arXiv:1011.4114](https://arxiv.org/abs/1011.4114).
- [40] R. C. Avohou, J. B. Geloun and M. N. Hounkonnou. Embedding Half-Edge Graphs in Punctured Surfaces (2017). [arXiv:1708.08720](https://arxiv.org/abs/1708.08720).
- [41] P. Elias, A. Feinstein and C. Shannon. A note on the maximum flow through a network. *IRE Transactions on Information Theory* **2**(4), pp. 117–119 (1956).
- [42] L. R. Ford and D. R. Fulkerson. Maximal Flow Through a Network. *Canadian Journal of Mathematics* **8**, pp. 399–404 (1956).
- [43] P. Hayden et al. Holographic duality from random tensor networks. *Journal of High Energy Physics* **2016**(11), p. 9 (2016). [arXiv:1601.01694](https://arxiv.org/abs/1601.01694).
- [44] R. Hartshorne. *Algebraic Geometry, Graduate Texts in Mathematics*, vol. 52. Springer New York, New York, NY. ISBN 978-1-4419-2807-8 (1977).
- [45] F. Verstraete and J. I. Cirac. Matrix product states represent ground states faithfully. *Physical Review B* **73**(9), p. 094423 (2006). [arXiv:cond-mat/0505140](https://arxiv.org/abs/cond-mat/0505140).
- [46] K. G. Wilson. The renormalization group: Critical phenomena and the Kondo problem. *Reviews of Modern Physics* **47**(4), pp. 773–840 (1975).
- [47] U. Schollwoeck. The density-matrix renormalization group in the age of matrix product states. *Annals of Physics* **326**(1), pp. 96–192 (2010). [arXiv:1008.3477](https://arxiv.org/abs/1008.3477).
- [48] S. Östlund and S. Rommer. Thermodynamic limit of density matrix renormalization. *Physical review letters* **75**(19), p. 3537 (1995).
- [49] J. Dukelsky, M. A. Martín-Delgado, T. Nishino and G. Sierra. Equivalence of the variational matrix product method and the density matrix renormalization group applied to spin chains. *EPL (Europhysics Letters)* **43**(4), p. 457 (1998).
- [50] G. Vidal. Class of Quantum Many-Body States That Can Be Efficiently Simulated. *Physical Review Letters* **101**(11), p. 110501 (2008). [arXiv:quant-ph/0610099](https://arxiv.org/abs/quant-ph/0610099).
- [51] O. Catoni. *Statistical Learning Theory and Stochastic Optimization, Lecture Notes in Mathematics*, vol. 1851. Springer Berlin Heidelberg, Berlin, Heidelberg, 1 edn. ISBN 978-3-540-22572-0 (2004).

- [52] O. Bousquet, S. Boucheron and G. Lugosi. Introduction to statistical learning theory. Summer School on Machine Learning, pp. 169–207. Springer (2003).
- [53] T. Hastie, R. Tibshirani and J. Friedman. The Elements of Statistical Learning. Springer Series in Statistics. Springer New York, New York, NY, second edn. ISBN 978-0-387-84857-0 (2009).
- [54] P. Mehta et al. A high-bias, low-variance introduction to Machine Learning for physicists. *Physics Reports* **810**, pp. 1–124 (2019). [arXiv:1803.08823](#).
- [55] A. Karpathy. CS231n Convolutional Neural Networks for Visual Recognition. <http://cs231n.github.io/convolutional-networks/>. (visited 2019-07-02).
- [56] R. Kindermann. Markov random fields and their applications. American mathematical society. ISBN 0-8218-5001-6 (1980).
- [57] D. Erhan, A. Courville, Y. Bengio and P. Vincent. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research* **11**(Feb), pp. 625–660 (2010).
- [58] R. Salakhutdinov and G. Hinton. Deep Boltzmann machines. Artificial intelligence and statistics, pp. 448–455 (2009).
- [59] B. Schölkopf and A. J. Smola. Learning with Kernels : Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, US. ISBN 9780262256933 (2001).
- [60] D. Liu et al. Machine learning by unitary tensor network of hierarchical tree structure. *New Journal of Physics* **21**(7), p. 073059 (2019). [arXiv:1710.04833](#).
- [61] Z.-Y. Han et al. Unsupervised Generative Modeling Using Matrix Product States. *Physical Review X* **8**(3), p. 031012 (2018). [arXiv:1709.01662v3](#).
- [62] I. Glasser, N. Pancotti and J. I. Cirac. Supervised learning with generalized tensor networks pp. 1–14 (2018). [arXiv:1806.05964](#).
- [63] N. Cohen, O. Sharir and A. Shashua. On the Expressive Power of Deep Learning: A Tensor Analysis. 29th Annual Conference on Learning Theory, pp. 698–728 (2015). [arXiv:1509.05009](#).
- [64] Y. Levine, O. Sharir, N. Cohen and A. Shashua. Quantum Entanglement in Deep Learning Architectures. *Physical Review Letters* **122**(6), p. 065301 (2019). [arXiv:1803.09780](#).
- [65] B. Poole et al. Exponential expressivity in deep neural networks through transient chaos. Advances in neural information processing systems, pp. 3360–3368 (2016). [arXiv:1606.05340](#).
- [66] M. Raghu et al. On the Expressive Power of Deep Neural Networks. Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 2847–2854. JMLR. org (2016). [arXiv:1606.05336](#).
- [67] R. Eldan and O. Shamir. The Power of Depth for Feedforward Neural Networks. Conference on learning theory, pp. 907–940 (2015). [arXiv:1512.03965](#).
- [68] Y. Bengio and O. Delalleau. On the expressive power of deep architectures. International Conference on Algorithmic Learning Theory, pp. 18–36. Springer (2011).

-
- [69] S. Cheng, J. Chen and L. Wang. Information Perspective to Probabilistic Modeling: Boltzmann Machines versus Born Machines. *Entropy* **20**(8), p. 583 (2018). [arXiv:1712.04144](#).
- [70] D.-L. Deng, X. Li and S. Das Sarma. Quantum Entanglement in Neural Network States. *Physical Review X* **7**(2), p. 021021 (2017). [arXiv:1701.04844](#).
- [71] J. Chen et al. Equivalence of restricted Boltzmann machines and tensor network states. *Physical Review B* **97**(8), pp. 1–18 (2018). [arXiv:1701.04831](#).
- [72] Z.-A. Jia et al. Entanglement Area Law for Shallow and Deep Quantum Neural Network States (2019). [arXiv:1907.11333](#).
- [73] R. G. Melko, G. Carleo, J. Carrasquilla and J. I. Cirac. Restricted Boltzmann machines in quantum physics. *Nature Physics* **15**(9), pp. 887–892 (2019).
- [74] I. Glasser et al. Neural-Network Quantum States, String-Bond States, and Chiral Topological States. *Physical Review X* **8**(1), p. 011006 (2018). [arXiv:1710.04045](#).
- [75] S. R. Clark. Unifying neural-network quantum states and correlator product states via tensor networks. *Journal of Physics A: Mathematical and Theoretical* **51**(13), p. 135301 (2018). [arXiv:1710.03545](#).
- [76] O. Sharir et al. Deep autoregressive models for the efficient variational simulation of many-body quantum systems (2019). [arXiv:1902.04057](#).
- [77] O. Sharir and A. Shashua. On the Expressive Power of Overlapping Architectures of Deep Learning. International Conference on Learning Representations (2018). [arXiv:1703.02065](#).

