University of Padua

Department of General Psychology

Bachelor's Degree Course in Psychological Science

Structure Development in Dynamically Pruned Artificial Neural Networks

Supervisor

Dr. Alberto Testolin, *Department of General Psychology*
University of Padua

Co-Supervisor

Prof. Samir Simon Suweis, *Department of Physics and Astronomy*
University of Padua

Candidate

Simone Betteti

ID

1138647

To my family, who has faith in my abilities, vision, and has always supported me without compromises.

Abstract

The human brain is a highly sophisticated, parallel architecture processing continuum-domain experiences in a discrete manner. From a microscopic point of view, it is the result of the local interactions among neurons belonging to a specific cortical column, whereas from a macroscopic point of view, it is that of the inter-dependencies among different macroscopic cortical regions. However, we as humans are often oblivious to the intricacies characterizing even the most elementary forms of cognition, presumably because of the spontaneity with which we perform most everyday tasks from a very early age. Indeed, the existing gap between the subjective domain of personal experiences and the objective neuroscience domain fosters reliance on concepts such as *'mind'*, *'unconscious'*, and the like, which although able to provide seemingly plausible explanations for complex human phenomenons fail to consolidate a reliable predictive model, displaying the level of inferential accuracy typical of most models in well established sciences such as Physics and Chemistry. It was in response to this issue, and also to the emergent curiosity about the possibility and feasibility to implement intelligent behaviour in machines, that researchers in the second half of the past century began to theorize more formal paradigms that could explain and reproduce information-processing capabilities peculiar to humans. Thus the race for the creation of an Artificial Intelligent system began.

In 1958, the entity today renowned as Perceptron, based on the Artificial Neuron proposed by McCulloch W., and Pitts. W., was formally introduced by Frank Rosenblatt [4]. He envisioned a simple pre-post synaptic system as a scalar field receiving binary or continuous input vectors describing certain aspects of the environment (set containing all instances of a specific class) and producing a scalar output. This reflects the encoding of features according to specific adaptive needs. Coherently, the ability to adapt was, and still is (albeit thoroughly revisioned), based on an error minimization process, known as *error back-propagation*, which assesses the discrepancy between the actual and expected output. Despite the initial hype, largely due to the model being able to solve some trivial classification problems, the momentum that research in the field had gain died almost abruptly. The reasons behind this collapse was the publication of the book *'Perceptron: An Introduction to Computational Geometry'* by Minsky M.L. and Papert S.A.[15], in which the authors formally demonstrated the limits of the Perceptron.

The shattering publication by Minsky and Papert allegedly caused a transition from a more biological approach, namely that of neural networks, to research in symbolic artificial intelligence, which relies on abstract representations, that lasted for about 20 years; and much as drastically as research froze, it also rekindled. The publication of the article *'Learning representations by back-propagating errors'* [17], as well as independent work by Le Cunn

(published in french), generated a burst of renewed interest in Artificial Neural Networks (ANNs) that, coupled with the exponential improvements of machine's hardware, made the acknowledgement of their potential for application possible. Since then, there has been a massive burst in the quantity and quality of the learning algorithms available, which vary in their ability to perform very narrow tasks depending on their architectural and functional properties. Despite such advances, most of this research has focused solely on the functional properties of ANNs, and specifically on how the introduction of additional parameters aids networks at learning complex representation in high-dimensional spaces through the exploration of the representation landscape.

In this thesis, I will address the issue of representation space exploration from a strictly topological perspective, using biological networks as paradigms to model synaptic development. The study of the dynamic evolution of a neural network may reveal emergent properties primary in the development of more complex and functionally powerful architectures. Indeed, the hope is to discover a tight correlation between the topological organization of the network and its capability to perform complex computations under connectivity constraints. Hence, the aim is to produce networks that display great computational power while having a significantly high number of inactive connections.

Other than being more biologically plausible than their fully connected, canonical analogues, such networks also display the potential for industrial applications in virtue of their sparseness. If correctly optimized, sparse architectures can drastically reduce the energy and time costs, thus enabling successful deployment in real-world scenarios (such as self-driving cars, where swift responses are vital). Finally, and perhaps more importantly, the real aim of this thesis is to plant a seed for the development of a comprehensive and exhaustive framework of intelligence. The analysis of the emergent structures and related properties of a developing network may help to define the building blocks of scalable architectures that, ultimately, could display human-like general cognitive abilities.

Contents

1

Neuronal Development

1.1 Neuronal Formation and Migration

The process of neuronal developments begins in the early weeks of gestation, specifically about 18 days after ovulation, with the formation of the neural groove, and the related neural plate. These structures will later develop and partition into the different components of the nervous system, with the process being almost completely governed by genetic instructions [7]. The neural groove then rounds and forms the neural tube, the rostral portion of which later divides into the two hemispheres and related cavities (lateral and midline third ventricle). These cavities are primary in the process of brain formation because from the germinal matrix, a region adjacent to them, most of the billions of neurons constituting it will later form, along with supporting cells such as astroglia and the oligodendroglia. The early partitioning between information processing units, namely neurons, and supporting units, glia cells, is primary in the determination of the infant brain architecture, which must be

fully equipped to successfully deal with the massive amount of input from the environment. Among the supporting units, radial glia play an essential role in the structural development of the brain, as postmitotic neurons migrates, either individually or in small communities, along the length of their radii. The process is coordinated by Cajal-Retzius cells, which seems to provide guidance in the positioning of neuronal clusters, as well as form a lower bound on neuronal migration, and ends around the 25^{th} week of gestation, with most of the neurons already formed and displaced.

The migration process produces a hierarchical structure, with the cerebral cortex being partitioned into six stratified layers, each predominantly characterized by the presence of a specific type of neuron; however, the discussion of these different types is beyond the scope of this thesis. One of the cardinal assumption behind the development of the new framework, formally introduced in chapter 4, is that despite the differences in size of individual layers, there seems to exist a kind of homogeneity in the way in which the cerebral cortex is functionally organized. Indeed, it seems to have a vertical, columnar organization, both in structural and functional terms; the functional segregation of columns has led to the formulation of the hypothesis of modular processing, according to which each of these different columns encodes a very narrow and specific feature, with adjacent columns supposedly encoding slight variations of the same feature [24]. One fundamental consequence of such homogeneity is the unlikeness of relating the functional differences of brain regions to the prevalence of a specific type of neuron. Instead, it is the same input to a specific cortical areas, in all its qualitative and quantitative variations, that finally determines the tuning of specific cortical circuits. Hence, it should be possible to theorize a basic processing unit that finely adapts to different qualities of input and that, if placed within a broader, complex system should be able to display higher cognitive functions as emergent properties; however, the building blocks of such a unit must be similar to a neuron.

A neuron is a biological information processing unit characterized by a body, called the soma, an elongated branch, the axon, which propagates electrical signals forward, and an arboreal

structure collecting the input of a finite number of other neurons. As units, neurons are not, depending on their location identity in the cortex, fully developed. Indeed, their structure undergoes continuous changes due to both environmental factors, such as normal input or exposure to degrading agents, and genetic ones, as is the myelination process, during which the axon is shielded by a myelin sheath, opportunely formed and provided by Schwann cells in the central nervous system (CNS) (see Fig.1.1).

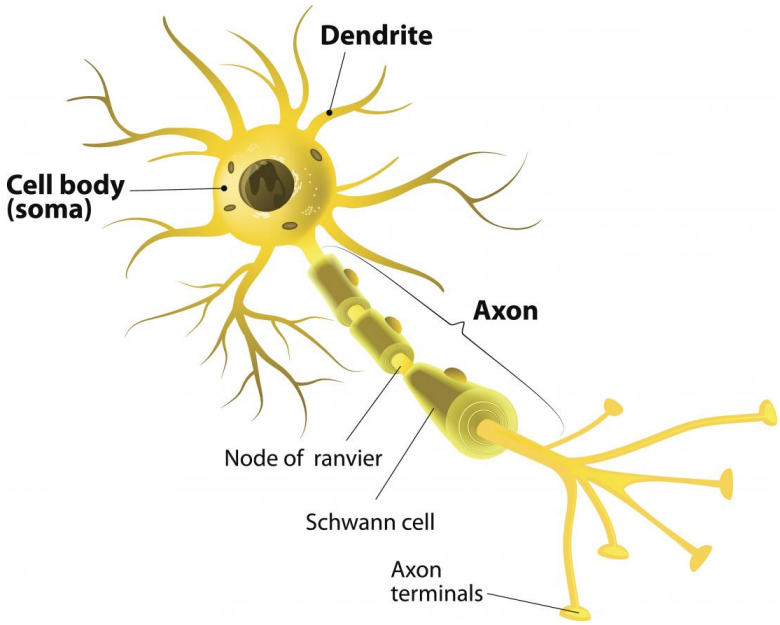


Figure 1.1: Structural representation of a neuron and its composing units.
https://www.medicalnewstoday.com/articles/320289.php

The myelin sheath significantly enhances the speed of signal transmission, and can finish maturing even in the 21^{st} year of age for neurons in the prefrontal cortex. Nonetheless, the structural modifications that hold the greatest value to the present thesis are those taking place on the dendrites of the neuron; specifically, the process of synaptic pruning, by which a massive number of synapses are suppressed.

1.2 Synptogenesis

A synapse is the site of 'contact' between two different neurons, where the actual relay of information, in the form of electrical or chemical signals (Fig.1.2), takes place.

Figure 1.2: Computer simulation of a chemical synapse

Contrary to axonal and dendritic growth, which are largely accounted for by genetic factors, synaptogenesis, that is the process of synapses formation, is mainly environment-driven, with the initial distribution of synapses being based on a proximity metric. It is likely that this random connectivity lay at the base of cortical circuitry specialized in learning from the environment, whereas more structured dynamics of synaptogenesis, such as bidirectional signalling - independent from external input - are more common in cortical areas responsible for less sophisticated but more vital functions. Perhaps surprisingly, infancy and early childhood are the life stages in which the highest synaptic density is registered, readily followed by a period of steady decline during which only the connections most functional to a specific circuitry are retained. It is therefore useful to distinguish between synaptic burst, the process starting at birth and culminating at about the second year of life during which the number

of existing synapses rapidly increases, and synaptic pruning, which is the subsequent period of synapses elimination that continues throughout childhood and adulthood. As stated by Hutterlocher [11], it is interesting to notice that the fact that synapses are much more densely packed on the dendrites of cerebral cortical neurons of infants and young children than those of adults may account for the increased excitability of the cerebral cortex in the young child. Of particular interest is the process of synaptogenesis in the visual cortex (area 17), characterized by the 4^{th} neuronal layer receiving the massive input from the two retinas through the lateral geniculate nucleus (LGN) and relaying the encoded signal to higher cortical areas. The synaptic density swiftly increases between the second and fourth month after birth, to then stabilize and gradually decrease to adult levels (see Fig.1.3)

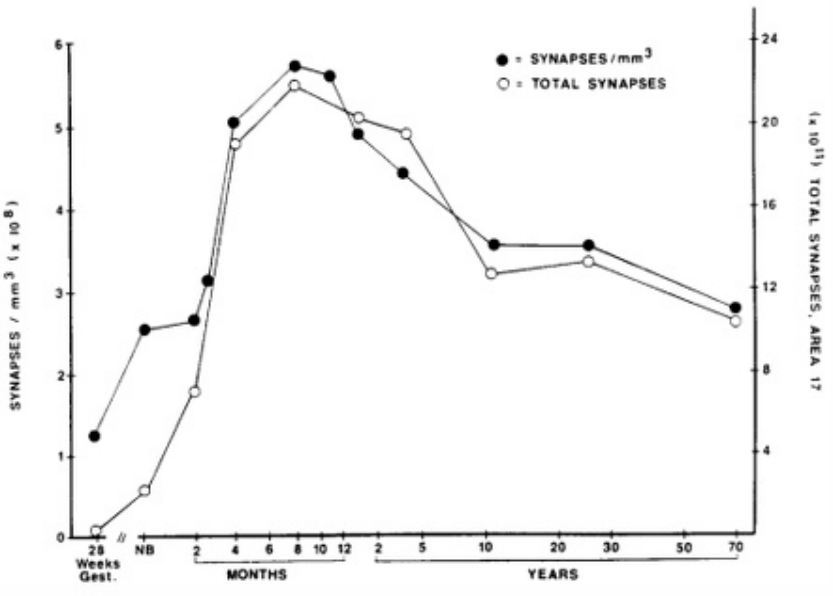


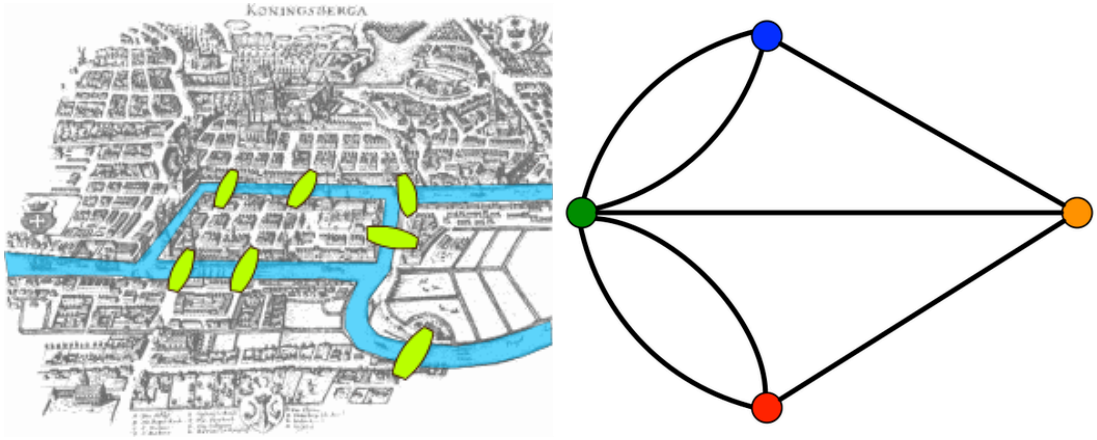
Figure 1.3: Synaptic density in the visual cortex throughout life. Courtesy of Hutterlocher P. R. (2002)

It is worth stating that synaptic pruning seems to be the heaviest in species such as humans and monkeys.

2

Graph Theory

Graph theory is one of the fundamental pillars of discrete mathematics, and its origin can be traced back to Euler's solution of the Königsberg bridge problem in 1735 (see Fig.2.1). Specifically, the problem required understanding whether it was possible to devise a walk through the city, comprehensive of a central island connected to the mainland by seven bridges, so that the walker would have to walk across each bridge only once. Euler reached his solution by schematizing the city map in a much simpler form, today known as graph.



(a) Map of the city **(b)** Euler's solution

Figure 2.1

Graphs are 1-dimensional geometric objects composed of nodes and edges that instantiate relationships of some sort between their elements. Formally, a graph is identified by a pair $G = (V, E)$ [5], where V is the set of vertices (or nodes) and E is the set of edges connecting any two elements of V. The graph is considered directed if E contains ordered sets of vertices $v \in V$ such that $\forall v_i, v_j \in V$, E may contain both $\{v_i, v_j\}$ and $\{v_j, v_i\}$; instead, the graph is undirected if and only if E is an unordered set of edges.

A graph is complete if $\forall v_i, v_j \in V \; \exists \{v_i, v_j\} \in E$, thereby if every node in V is connected to every other node, and is identified with K_n, with $n =\mid V \mid$. Finally, a special subclass of graphs are the bipartite graphs [23], defined as $G = (U, V, E)$, where U and V are subset of nodes such that $\forall u_1, u_2 \in U \nexists \{u_1, u_2\} \in E$ and $\forall v_1, v_2 \in V \nexists \{v_1, v_2\} \in E$, and the only edges allowed are of the type $\{u_i, v_j\}$, $i = 1, .., n, j = 1, .., m \wedge n =\mid U \mid, m =\mid V \mid$. If the bipartite graph is also complete, then it is defined as $K_{n,m}$.

In order to evaluate if two nodes $y, z \in V$ are connected, we assess the existence of a *Path*, which is a set of edges $L = \{\{y, x_1\}, .., \{x_\mu, z\}\}$ indirectly connecting the two, with $\mu =\mid L \mid$ being the length of the path.

2.1 Complex Systems

From the beginning of the XX century graphs have been used in a variety of context to map the interaction between different agents, and only in the past decades have acquired a fundamental role in the study of complex systems (see Fig.2.2) such as power grids, climate, and the human brain.

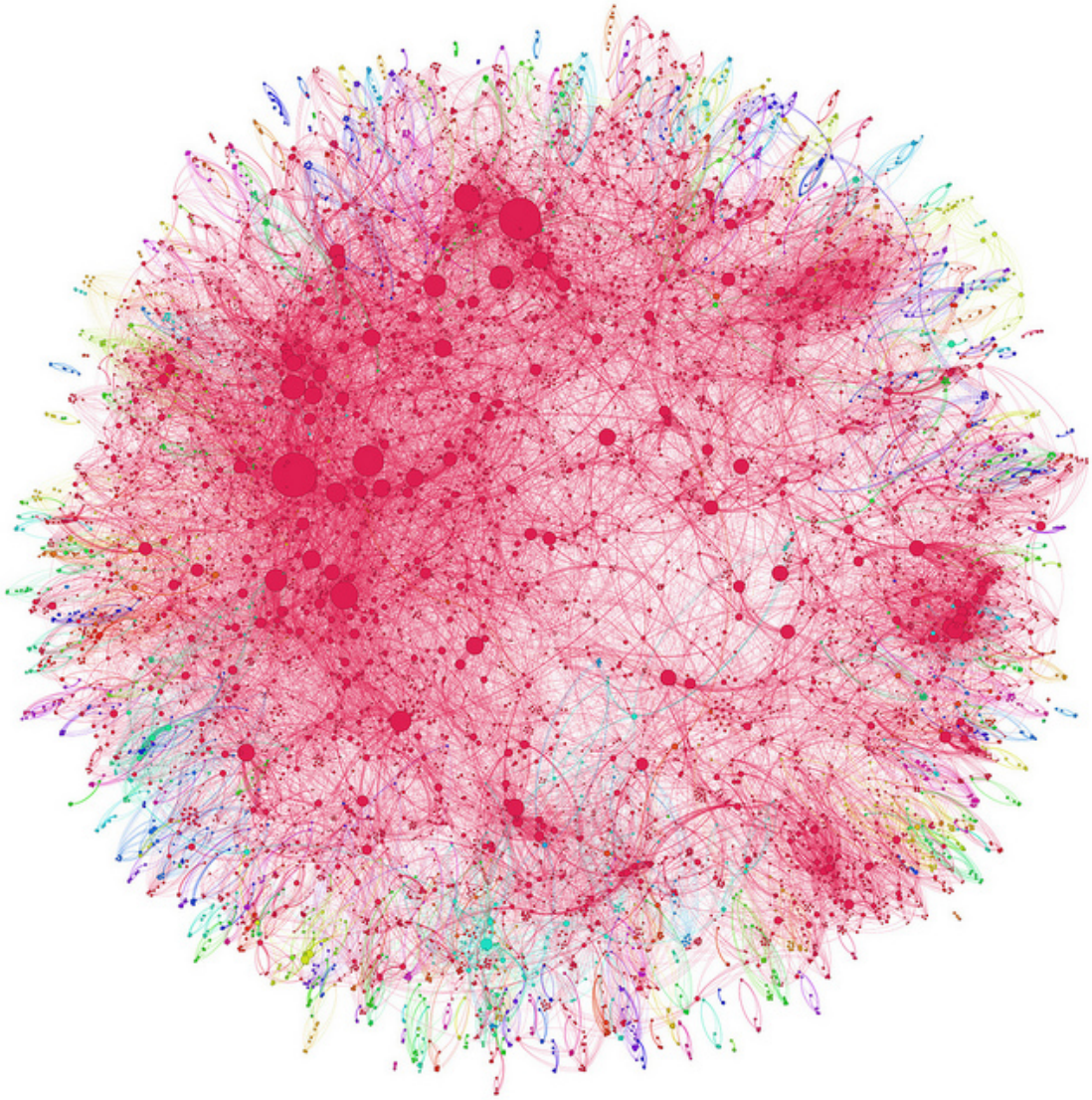


Figure 2.2: Graphic visualization of a complex network

More broadly, the study of complex systems is aimed at predicting the behaviour of a system governed by multiple interacting variables, represented by nodes, and at the computation of well-defined measures that describe the state of the system at a specific moment. A subset of this measures are the so called centrality measures, which, in simple terms, try to establish the relevance of each node in the network according to some criterion. The present thesis has relied on three of these, namely Degree Centrality, Eigen Centrality, and Betweenness Centrality.

2.1.1 Degree Centrality

The degree centrality measure returns the importance of each node in the network according to its degree, which corresponds to the number of edges that are incident on it. The degrees of the vertices in most networks are highly right-skewed [16]. Additionally, it is possible to define the probability p_k that a node has degree k, which trivially corresponds to the fraction of n nodes in the network having degree k.

$$p_k = \begin{cases} 0 & if \quad deg(v_i) \neq k \\ p_k + \frac{1}{n} & otherwise \end{cases}$$

2.1.2 Eigenvector Centrality

Eigenvector centrality measures the influence that individual nodes have on the network [6]; precisely, the influence of a node depends on how many other high influence nodes are connected to it. Formally, given x an ordinary node in the network, its Eigen centrality score $C_E(x)$ is initialized as 1 and updated according to

$$C_E(x) = \frac{1}{\lambda} \sum_{y \to x} C_E(y) \tag{2.1.1}$$

which can also be written as

$$\lambda e = Ae \tag{2.1.2}$$

where λ is the greatest eigenvalue of the Adjacency matrix A of $G = (V, E)$. The solution of the equation yields the eigenvector e, the i^{th} coefficient of which gives the Eigen Centrality score for the i^{th} node in the graph. If the graph has weighted edges, each edge with weight *w*, then the updating rule becomes

$$C_E(x) = \frac{1}{\lambda} \sum_{y \to x} w \cdot C_E(y) \tag{2.1.3}$$

2.1.3 Betweenness Centrality

The Betweenness Centrality can be thought of as a measure of the importance of a node at the efficient relay of information across the network. Specifically, for each fixed node x in the graph, the Betweeness score $C_B(x)$ gives the number of geodesic connecting two generic nodes $y, z \in V,\ y \neq z \neq x$, where a geodesic is the shortest path between y and z. Formally

$$C_B(x) = \sum_{x \neq y \neq z \in V} \frac{\sigma_{yz}(x)}{\sigma_{yz}} \tag{2.1.4}$$

with σ_{yz} being the total number of shortest path between y and z and $\sigma_{yz}(x)$ is the number of those that pass through x.

2.1.4 Clustering Coefficient

Finally, the use of the statistics for the clustering coefficient, or curvature, of each node in the graph has allowed for a first assessment of the topological evolution of the network in terms of formation and disruption of $1 - loops$, or triangles. Specifically, this measure aims at establishing whether, given a node $x \in V$ and $y, z \in V/\{x\}$ and that $\exists\{x, y\}, \{x, z\} \in$

$E \to \exists\{y, z\} \in E$, which gives the score

$$C_X = \frac{\zeta_x}{\zeta} \tag{2.1.5}$$

where ζ_x is the number of triangles connected to node x and ζ the number of triplets (path of length 3) centered on node x. Equivalently, given ν the neighbourhood of node x (set of nodes connected to x through a path of length 1)

$$C_X = \frac{2\zeta}{\nu \cdot (\nu - 1)} \tag{2.1.6}$$

The clustering coefficient for the whole network thus becomes

$$C = \frac{1}{n} \sum_X C_X \tag{2.1.7}$$

with $n = |V|$.

Accordingly, if the graph has weighted connections it is possible to define a weighted clustering coefficient

$$WC_X = \frac{1}{\nu \cdot (\nu - 1)} \sum_{y,z \in V} (w_{x,y} + w_{x,z}) \cdot (a_{x,y} a_{x,z} a_{y,z}) \tag{2.1.8}$$

as defined in [3], where

$$a_{x_1,x_2} = \begin{cases} 1 & if \ \{x_1, x_2\} \in E \\ 0 & otherwise \end{cases}$$

3

Generative Models

3.1 Boltzmann Machines (BMs)

Boltzmann machines (BMs) [1] identify a class of machine learning algorithms specialized in the task of unsupervised learning, therefore capable of creating powerful internal representations of the provided data, that directly stem from an expansion of Hopfield's associative networks [10]. Their basic functioning requires an initial phase in which the training data are clamped onto the model units and from which a training procedure is routinely instantiated, with the value of each unit dependent on its Markov Blanket [13], which is defined as the set of neighbouring units.

The name Boltzmann machine formally refers to a probability distribution defined over a binary vector $x \in \mathbb{K}^n,\ \mathbb{K} = \{0, 1\}$ governed by the law

$$P(x) = \frac{1}{Z} e^{-E(x)} \tag{3.1.1}$$

where the scalar field $E : \mathbb{K}^n \to \mathbb{R}$ is the energy function

$$E(x) = -\sum_{i<j} x_i w_{i,j} x_j \tag{3.1.2}$$

and Z is the partition function $Z = \sum_x e^{-E(x)}$ ensuring that $P(x)$ returns a valid probability value. The energy function imposes specific constraints, such that highly probable instances in the data are associated with energy minima on the energy surface [12], but it does not lend itself well for sampling purposes and, for this reason, approximate procedures such as Gibbs Sampling or Metropolis algorithm are needed.

Within the formalism of Graph Theory, it is possible to envision a BM as a unipartite graph $G = (V, E)$ (see Fig.3.1), with $n = |V|$ and the existence of edges, either weighted or not, between nodes specified by a connectivity matrix $W \in M_n\{\mathbb{K}\}, \mathbb{K} = \{\mathbb{R}, \{0, 1\}\}$ where

$$w_{i,j} = \begin{cases} 0 & if (i,j) \notin E \\ k & otherwise \end{cases}$$

with $k \in \mathbb{K}/\{0\}$. It is interesting to note that in the case of an undirected graph, the connectivity matrix W is symmetric (i.e. $w_{i,j} = w_{j,i}$). The absence of an edge between two units implies an important statistical property, which is primary in the creation of simpler but incredibly powerful models; specifically, given three units (or nodes) *A*, *B*, and *C*, if $w_{A,B} \neq 0$, $w_{B,C} \neq 0$ but $w_{A,C} = 0$, then the values over A and C are conditionally independent given the value of B (Markov Property). Thus, by extending the absence of connections among all units belonging to a specific set, with x_i , $i = 1, .., k$ being the value associated to each unit in this set, and x_b the value of *B* not belonging to the set and having connection with all the units in it, it holds

$$P(x_1, .., x_i, .., x_k | x_b) = \prod_{i=1}^{k} P(x_i | x_b) \tag{3.1.3}$$

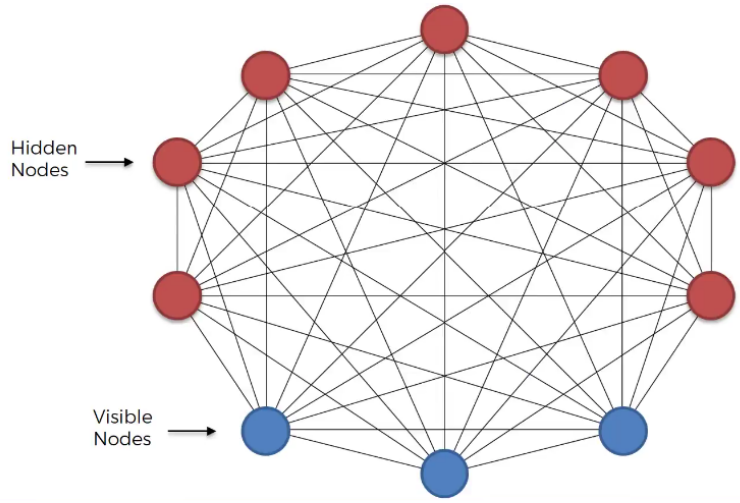


Figure 3.1: Graphic Representation of a Boltzmann Machine,
https://medium.com/datadriveninvestor/an-intuitive-introduction-of-boltzmann-machine-8ec54980d789

If instead we define a fully connected architecture, the probability distribution over its nodes would not factorize so neatly and, more importantly, will plummet into recursion of values estimation for each unit. Consequently, in 1986 Paul Smolensky [19] exploited the Markov property to generate a new type of information processing algorithm, at the time called Harmonium, which has its core properties identical to those of a Restricted Boltzmann Machine.

3.2 Restricted Boltzmann Machines (RBMs)

The general version of the Boltzmann Machine has only been rarely studied due to the extremely high computational costs associated to its training, and as a result focus has shifted from this supposedly more powerful model to a *'Restricted'* version, named Restricted Boltzmann Machine (RBM). RBMs are probabilistic graphical models with a bipartite architecture $K_{n,m}$ (see Fig.3.2), which differently from the more general BMs stratify units hierarchically and elides inter-dependencies among the units belonging to the same layer. This type of algorithm has proven to be incredibly well suited for the reproduction of images and sounds, as well as for signal denoising [20].

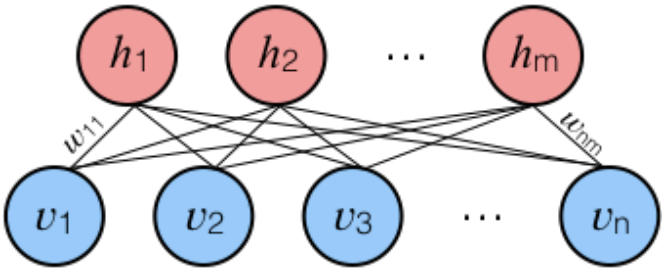


Figure 3.2: Graphic representation of a Restricted Boltzmann Machine

Formally, we define the energy function of the RBM as

$$E(V, H) = -V^T W H - V^T b_V - H^T b_h \tag{3.2.1}$$

with $V \in \mathbb{R}^n$ being the vector representing the state of each unit in the visible layer, $H \in \mathbb{R}^m$ the vector representing the state of each unit in the hidden layer, $W \in M_{n,m}$ the connectivity matrix, and $b_v \in \mathbb{R}^n$, $b_h \in \mathbb{R}^m$ the biases over the visible and hidden layers, respectively, which provide a firing offset. It is worth noticing that, from a biological standpoint, the weights specified by the connectivity matrix report quantitative information on the strength of the individual between-layers connections.
The new probability distribution defined over the RBM is

$$P(V, H) = \frac{1}{Z} e^{-E(V,H)} \tag{3.2.2}$$

and, depending on the type of RBM - binary or continuous valued - the partition function Z becomes

$$Z = \sum_{V \in \{0,1\}^n} \sum_{H \in \{0,1\}^m} e^{-E(V,H)} \tag{3.2.3}$$

or

$$Z = \int_{J \subseteq \mathbb{R}^n} \int_{K \subseteq \mathbb{R}^m} e^{-E(V,H)} \tag{3.2.4}$$

which most of the time happens to be intractable.

By exploiting the above mentioned Markov Property, it is straightforward to observe that

$$P(H|V) = \prod_{j=1}^{m} P(H_j|V) \tag{3.2.5}$$

$$P(V|H) = \prod_{i=1}^{n} P(V_i|H) \tag{3.2.6}$$

with the actual probability being delivered by a Sigmoid function

$$P(H_i = 1|V) = \frac{1}{1 + e^{V^T W_{.i}}} \tag{3.2.7}$$

For computational reasons. it is more feasible to consider the log-probability instead of the 3.2.5 and 3.2.6, since

$$log\ P(H|V) = log \prod_{j=1}^{m} P(H_i|V) \tag{3.2.8}$$

$$= \sum_{i=1}^{m} log\ P(H_i|V) \tag{3.2.9}$$

produces a finite sum instead of a finite multiplication; for analogous reasons, we consider

$$logP(V, H) \tag{3.2.10}$$

The actual learning of a data distribution involves 'navigation' of the energy landscape [21], with the aim of finding the set of weights and biases W, b_v, b_h that best capture the most salient features in the data, and thereby those that are able to furnish its best parametrization. Computing the error, by reliance on gradient descent, on 3.2.10 often proves unfeasible, and for this reason the learning algorithm relies on a stochastic gradient ascent procedure named 'Contrastive Divergence (CD)', which differentiates between a positive and negative training phase. In the positive phase, the system is said to be 'data-driven', as it propagates

its activation, from the clamped data vector, to the hidden representation vector. Instead, in the negative phase the system is described as 'model-driven', as the data vector is unclamped and the signal is propagated from the new hidden, representational, layer to the visible layer, and then back to the hidden layer. The two phases are followed by the parameters update procedure, in which agreement between data-driven visible and hidden states and the related model-driven ones is maximized. Specifically

$$\frac{\partial \, log \; P(V, H, b_v, b_h)}{\partial W} = \mathbb{E}(VH^T)_{data} - \mathbb{E}(VH^T)_{model} \tag{3.2.11}$$

$$\frac{\partial \, log \; P(V, H, b_v, b_h)}{\partial b_v} = \mathbb{E}(V)_{data} - \mathbb{E}(V)_{model} \tag{3.2.12}$$

$$\frac{\partial \, log \; P(V, H, b_v, b_h)}{\partial b_h} = \mathbb{E}(H)_{data} - \mathbb{E}(H)_{model} \tag{3.2.13}$$

and the related update of parameters is

$$\Delta\omega = \epsilon \frac{\partial \, log \; P(V, H, b_v, b_h)}{\partial \omega} \tag{3.2.14}$$

where $\omega \in \{W, b_v, b_h\}$ and $\epsilon \in (0, 1)$.

One of the peculiar properties of contrastive divergence is that, if implemented with L_2 regularization,

$$\Delta CD_\lambda(W) = \Delta CD(W) - \lambda W$$

always has a set of fixed points, as formally demonstrated by Tijmen and Sutskever [20], and therefore guarantees exact reproduction, through the use of a suitable parametrization, of the data-distribution. Specifically, the following theorem holds

Theorem 1 *For any binary RBM of any architecture and any* $\lambda \in (0, 1)$, $\exists W* \in M_{n,m}\{\mathbb{R}\}$: $\Delta CD(W*) = 0$.

Although such convergence is theoretically possible, it often proves unnecessarily expensive to compute, and therefore approximations are used instead. It should be now clear the structural and functional analogy existing between an RBM and a cortical column, to the extent where the former may be seen as a simplification of a vertical section of the latter.

3.3 Deep Belief Networks (DBNs)

Deep Belief Networks [9] are a class of generative models creating representation of the input through subsequent encoding in multiple hidden layers. Just as the name suggests, DBNs are nothing else than multiple RBMs stack on top of each other (see Fig.3.3); their training is, differently from models exploiting back-propagation, layer-centered, meaning that the single RBMs are individually and sequentially trained using as input the representation created by the previously trained RBM (or the input vector if we consider the first RBM).

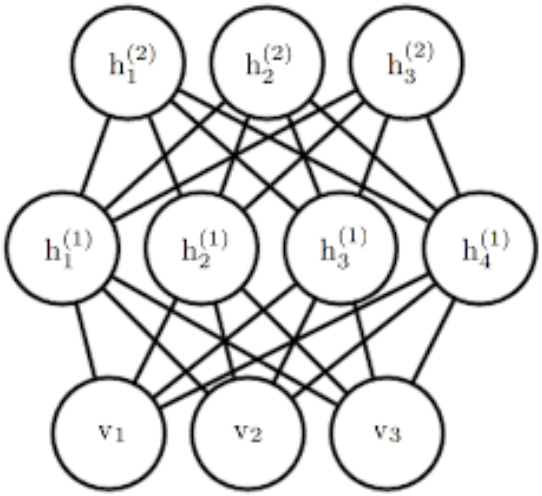


Figure 3.3: Graphic representation of a Deep Belief Network

Other deep generative models, which contrary to DBNs do not learn representation in a greedy manner, are Deep Boltzmann Machines (DBMs) [18]; in DBMs' training, the probability distribution is computed over all the layers of the architecture.

4

ColumnarNet: Framework and Dynamics

The process of training a DBN with contrastive divergence generates weight matrices that in theory may have, especially if complemented with regularizing procedures, several zeroed coefficients. However, in most cases the resulting architecture is a fully connected $K_{n,m}$ graph [21], which reflects the quasi saturation of connections among units - considered as ideal neurons. It is crystalline how such a resulting architecture, other then being biologically implausible, is also scarcely efficient in computational terms. It is therefore useful to instantiate an architecture that, other then resulting in a more effective connectivity, also develops interesting topological properties as a result of training.

4.1 Weight Initialization

In a canonical DBN, the weights between each pair of layers are normally initialized and multiplied by a scaling factor, so that the resulting weights matrices $W \in M_{n,m}(\mathbb{R})$ contains almost exclusively values in the interval $[-1, 1]$, with almost all the coefficients $w_{i,j} \neq 0$. If

we take each weight matrix to symbolize the synaptic connectivity between two wide regions of different layers, the condition of full connectivity seems dauntingly excessive. Thus, Prof. Suweis and I hypothesized the existence of modules, possibly symbolizing adjacent cortical columns, which formalize the high connectivity of neurons that are close together during the phase of migration and that nonetheless retained a smaller number of connections with neurons belonging to different columns, hence creating a structural difference between intra- and extra-modular connections. More formally,

Def: *Given DN a Deep Belief Network, the RBM between the* i^{th} *and* $(i+1)^{th}$ *layers will be defined as* $DN_{i,i+1}$.

Let us then consider a single RBM within a DBN such that $RBM := DN_{h_i,h_{i+1}}$, with $\mid h_i \mid = n$ and $\mid h_{i+1} \mid = m$. Given q as a common divisor of both n and m, it is possible to create sets of subsets of units in h_i and h_{i+1}, V_i and V_{i+1} respectively, such that $|V_i| = |V_{i+1}| = q$. Hence, given $V_i = \{n_1, .., n_q\}$ and $V_{i+1} = \{m_1, .., m_q\}$, it is possible to consider each of the couples (n_s, m_s) for $s \in 1, .., q$ as an individual RBM processing only a fraction of the input. It follows then that the correspondent weight matrix for $DN_{h_i,h_{i+1}}$ will be

$$W = \begin{pmatrix} W_{n_1,m_1} & 0_{n_1,m_2} & \dots & 0_{n_1,m_q} \\ 0_{n_2,m_1} & W_{n_2,m_2} & \dots & 0_{n_2,m_q} \\ \vdots & \vdots & \ddots & \vdots \\ 0_{n_q,m_1} & \dots & \dots & W_{n_q,m_q} \end{pmatrix}$$

with $W_{n_j,m_j} \in M_{n_j,m_j}(\mathbb{R})$ the weight matrix of a single module.
I have thus defined the core of our sparse weight matrix for $DN_{h_i,h_{i+1}}$. It is now necessary to proceed by defining the extra-modular connections, which are, connections between $i \in n_s$ and $j \in m_t$ for $s \neq t$. This definition will rely upon a deliberately chosen probability p; it

was suggested that a possible criteria for such a choice could be the fraction of connections over the total number of possible connections $n \cdot m$ for which the diameter of the resulting graph decreases significantly, following a process of gradual addition.
I therefore create a new matrix W_ϕ, such that for any $W_{\phi_{i,j}}$ the probability of having a real value different from zero is exactly p, and of having value zero $(1-p)$. Subsequently, we silenced to 0 also all the coefficients $W_{\phi_{i,j}}$ such that the couple (i,j) belongs to one of the sets $n_s \times m_s$, so that at the end we will have a matrix

$$W_\phi = \begin{pmatrix} 0_{n_1,m_1} & \Phi_{n_1,m_2} & \dots & \Phi_{n_1,m_q} \\ \Phi_{n_2,m_1} & 0_{n_2,m_2} & \dots & \Phi_{n_2,m_q} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{n_q,m_1} & \dots & \dots & 0_{n_q,m_q} \end{pmatrix}$$

It would then be possible to define our final weight matrix for $DN_{h_i,h_{i+1}}$ as

$$W_f = W + W_\phi$$

$$W_f = \begin{pmatrix} W_{n_1,m_1} & \Phi_{n_1,m_2} & \dots & \Phi_{n_1,m_q} \\ \Phi_{n_2,m_1} & W_{n_2,m_2} & \dots & \Phi_{n_2,m_q} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{n_q,m_1} & \dots & \dots & W_{n_q,m_q} \end{pmatrix}$$

As this is just a weight matrix initialized through an uncommon procedure, it is straightforward to see that all the results found for ordinary RBMs hold also for the present case. It is now useful to distinguish between two different type of learning, namely *Static Learning* and *Dynamic Learning*, as algorithmically different processes yielding sparse connectivity

matrices able to maintain state-of-the-art accuracy, under appropriate modifications.

4.2 Static Learning

In order to constrain the architecture of the RBM to a specific a priori determined topology it is necessary to formalize the time dependence of the learning process. Consequently, it turns out to be particularly useful to define one epoch as our time unit, such that

$$1\ epoch = 1\ t = [T]$$

Hence, we have that

$$W_f(0) = W_f = \begin{pmatrix} W_{n_1,m_1} & \Phi_{n_1,m_2} & \dots & \Phi_{n_1,m_q} \\ \Phi_{n_2,m_1} & W_{n_2,m_2} & \dots & \Phi_{n_2,m_q} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{n_q,m_1} & \dots & \dots & W_{n_q,m_q} \end{pmatrix}$$

and, more generally

$$W_f(t) = \begin{pmatrix} W_{n_1,m_1}(t) & \Phi_{n_1,m_2}(t) & \dots & \Phi_{n_1,m_q}(t) \\ \Phi_{n_2,m_1}(t) & W_{n_2,m_2}(t) & \dots & \Phi_{n_2,m_q}(t) \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{n_q,m_1}(t) & \dots & \dots & W_{n_q,m_q}(t) \end{pmatrix}$$

The idea behind static learning is rather simple; namely, after each epoch, all the connections that possessed at $t = 0$ a real value different from zero ($W_f(0)_{i,j} \neq 0$) will retain their modified weight, whereas all the other units will be routinely silenced. It must be noted that this type of learning allows no network flexibility, and may therefore produce rather poor samples, on top of bearing very little similarity to actual biological neuronal networks.

4.3 Dynamic Learning

4.3.1 Dynamic Learning I

The dynamic learning algorithm starts with the same argument as its static counterpart, which is, $W_f(t)$ for $t = 0$, where we assume that p was sufficiently small to guarantee high sparsity of the weight matrix. Hence, given the strict modularity, we would expect modules to display a highly conservative behaviour, with the units in layer h_i connecting almost exclusively to units in their related module in h_{i+1}; however, our expectations may be proven wrong, and therefore it is useful to define a conservation coefficient $\varphi : \mathbb{N} \to \mathbb{Q}$ as

$$\varphi(W_f(t)_{i,j}) = \frac{in_i(t)}{in_i(t) + ex_i(t)} = \varphi_i(t) \tag{4.3.1}$$

where in_i is the number of intra-modular units to which $i \in h_i$ affers to and ex_i is the number of extra-modular units to which it affers to.
As we know, during the prenatal developmental period neurons form and then drift vertically into one of the six layers constituting the cortex. It may be plausible to suppose, in a rather simplistic manner, that if

- $\frac{in_i(t)}{in_i(t)+ex_i(t)} > \frac{ex_i(t)}{in_i(t)+ex_i(t)}$ the neuron has efficiently drifted with its module in its current location, and consequently the distance from the other intra-modular neurons is minimal
- $\frac{in_i(t)}{in_i(t)+ex_i(t)} < \frac{ex_i(t)}{in_i(t)+ex_i(t)}$ the neuron has inefficiently drifted apart from the other neurons in its module, and consequently the distance from the other alleged intra-modular neuron is maximal
- $\frac{in_i(t)}{in_i(t)+ex_i(t)} \sim \frac{ex_i(t)}{in_i(t)+ex_i(t)}$ the neuron is a bridge between two or more modules

It is now necessary to determine the role that this coefficient plays in morphing the topology of the network. Considering the strength of units in layer h_{i+1}, given for each epoch by

$$s_j(t) = \sum_{i=1}^{n} W_f(t)_{i,j} \tag{4.3.2}$$

and because the dynamics of synaptic connectivity in the developing brain, as seen in the subsection 1.2, depends strictly upon environmental input, it may be sensible to hypothesize that the degree $d_j(t+1)$ of each unit $j \in h_{i+1}$ at time $t+1$ should depend on the afferring strength, which modulates signal relay to the unit. Hence, by defining the average absolute strength of nodes $\{j_1, ..., j_m\}$ as

$$\mu_{s(t)} = \frac{1}{m} \sum_{j=1}^{m} \mid s_j(t) \mid \tag{4.3.3}$$

Thus, $\forall j \in h_{i+1}$ we hypothesize that the degree of each unit in layer h_{i+1} should be

$$d_j(t+1) = d_j(t) - [d_j(t) \cdot \alpha(s_j(t), \mu_{s(t)})]$$

with the assumption that alpha must naturally lead to zero the second term $[d_j(t) \cdot \alpha(s_j(t), \mu_{s(t)})]$ when $\mid s_j(t) \mid - \mu_{s(t)} \sim 0$; hence

$$\alpha(s_j(t), \mu_{s(t)}) = \frac{\lambda \cdot (\mid s_j(t) \mid - \mu_{s(t)})}{max_{j \in h_{i+1}}(\mid s_j(t) \mid)}$$

for $0 < \lambda < 1$. The basic idea behind this specific conceptualization is that, by considering the absolute value of the afferring strength, if

- $|s_j(t)| > \mu_{s(t)}$ then the neuron j has distributed its feedback neurotransmitters to too many afferring neurons is
- $|s_j(t)| < \mu_{s(t)}$ then the neuron j has distributed its feedback neurotransmitters to too few afferring neurons is
- $|s_j(t)| \sim \mu_{s(t)}$ then the neuron j has distributed its feedback neurotransmitters to an adequate number of afferring neurons is

This evaluation will give us information about the number of connections that must be active for each column in $W_f(t+1)$. As previously stated, contrastive divergence in 1 t catalyzes the deviance from sparsity of the whole weight matrix. This is coherent with the fact that, in general, as a result of the update rule, each and every coefficient in the matrix will gain a real value different from zero, although regularizers such as weight decay may theoretically plummet them to 0. Consequently, it is primary to determine $\forall j \in h_{i+1}$ which connections must be preserved and which instead must be silenced. To this purpose, we long for a function that taken as input the conservation coefficient of the afferring units and the weight of the connection returns the fitness of the connection for conservation. This conceptualization follows naturally from an adaptation of neuronal plasticity for biological systems, where the strength of the connection of two neurons by means of synapses determines either its preservation or its deterioration. The subsequent scalar field has the potential to deliver accurate estimates of *fitness for conservation*.

Theorem 2 *Given the weight matrix $W_f(t)$ and the conservation coefficient $\varphi_i(t)$, we define fitness for conservation as the scalar field $F_c : \mathbb{Q} \times \mathbb{R} \to \mathbb{R}$,*

$$F_c(\varphi_i(t), W_f(t)_{i,j}) = 2e^{\varphi_i(t)} \int_0^{|W_f(t)_{i,j}|} e^{-w^2}\, dw \; - \; (2\varphi_i(t) - 1)^2 \tag{4.3.4}$$

where the first term is strictly positive and the second gives greater importance to bridge neurons.

This fitness functions should be the least sensible to perturbations when the weight has large absolute value, therefore guaranteeing stability to those connection that had high strength even before the update procedure. Indeed, as clearly observable from 4.3.4, the only contributor to the overall increase in fitness value is $\int_0^{W_f(t)_{i,j}} e^{-w^2}\, dw$, and therefore is consistent with the survival of the strongest principle. Nonetheless, functional aspects of the first term are complemented by topological considerations on the adequacy of the neuron's position, as specified by $e^{\varphi_i(t)}$. However, it is also of uttermost importance to preserve the connections of bridge neurons; consequently, connections belonging to those units are the most likely

to survive, as they are the least penalized by the term $(2\varphi_i(t) - 1)^2$. It is important to use as weight matrix one on which the update following contrastive divergence is not blunted by learning factors, as this would seriously hinder any process of dynamic learning. As a marginal note, we compute, by supposing a continuous co-domain for the conservation coefficient and therefore $F_c(t) : \mathbb{R}^2 \to \mathbb{R}$.

$$\frac{\partial F_c(\varphi_i(t), W_f(t)_{i,j})}{\partial \varphi_i(t)} = 2e^{\varphi_i(t)} \cdot \varphi_i'(t) \int_0^{|W_f(t)_{i,j}|} e^{-w^2}\, dw \, - \, 4\varphi_i'(t)(2\varphi_i(t) - 1) \quad (4.3.5)$$

$$\frac{\partial F_c(\varphi_i(t), W_f(t)_{i,j})}{\partial W_f(t)_{i,j}} = 2e^{\varphi_i(t)} \cdot e^{-{W_f(t)_{i,j}}^2} \quad (4.3.6)$$

As it is clearly deducible from the second partial derivative, for fixed $\varphi_i(t)$ the regions of steeper slope are found for $w_{i,j} \to 0$, thus suggesting a general strive for the increase of the weights absolute value. Consequently, it is possible to create a fitness matrix having as coefficients the outputs of the fitness for conservation equation for each $W_f(t)_{i,j}$

$$F_c(t) = \begin{pmatrix} F_c(t)_{1,1} & F_c(t)_{1,2} & \dots & F_c(t)_{1,m} \\ F_c(t)_{2,1} & F_c(t)_{2,2} & \dots & F_c(t)_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ F_c(t)_{n,1} & F_c(t)_{n,2} & \dots & F_c(t)_{n,m} \end{pmatrix}$$

It is now desirable to update the weight matrix according to the established topology and synaptic modification rules. By taking into account the constraints imposed $\forall j \in h_{i+1}$, namely that their degree must be $d_j(t)$, $W_f(t)$ can be updated by means of the following algorithm

Algorithm 4.1 Forced Topology Algorithm

Result: $W_f(t)$

INPUT $W_f(t),\ d(t),\ F_c(t)$

for *j in size($W_f(t)$,2)* do

ranking=order($F_c(t)$(1:end,j));

ranking((d$_j(t)+1) : end) = 0$;

$preserve = index(F_c(t) == ranking)$;

$update = zeros(m, 1)$;

$update(preserve) = 1$;

W$_f(t)$(1:end,j)=$W_f(t)$(1:end,j).*update;

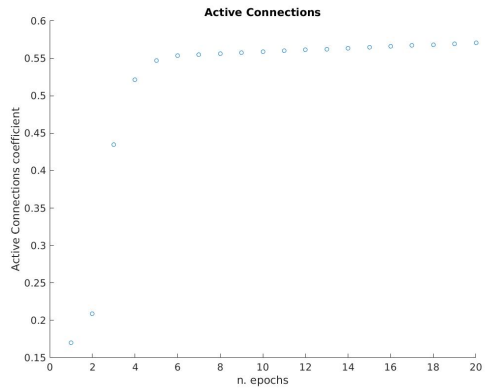
end

4.3.2 Dynamic Learning II

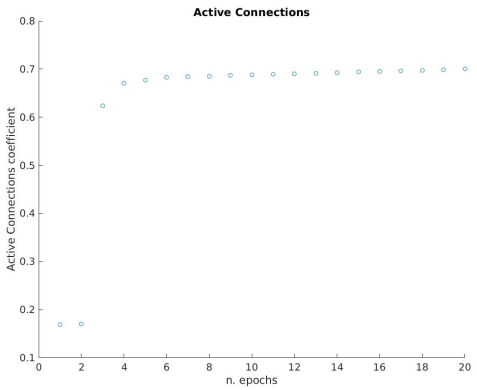
The first iteration of the algorithm *Dynamic Learning I* did not produce interesting results in terms of accuracy, as this was considerably below standards. However, by plotting the fraction of active connections, computed as

$$fA_c(t) = \frac{\sum_{j=1}^{m} d_j(t)}{n \cdot m} \tag{4.3.7}$$

where $d_j(t)$ is the degree of unit j for $j \in h_{i+1}$, it was immediately clear how the process displayed impressive initial sensitivity (see Fig.4.1) to the strength of the connections.



(a) Fraction of active connections for H1-H2



(b) Fraction of active connections for H2-H3

Figure 4.1

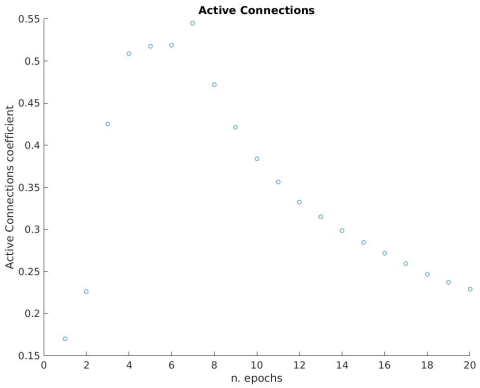
Accordingly, I opportunely modified the algorithm so that if after the sixth epoch the condition $A_c(t+1) - A_c(t) < \epsilon, \;\; \epsilon \in \mathbb{R}$ was satisfied, then the updating according to the *Dynamic Learning I* algorithm would stop. A form of threshold learning would then be abruptly implemented, for which if after Contrastive Divergence $W_f(t)_{i,j} < \delta, \;\; \delta = 0.0015 + 0.001t$, then $W_f(t)_{i,j} = 0$. Other than producing results in terms of accuracy that were equal if not better than those achieved by a canonical architecture, the resulting plot (see Fig.4.2) of the fraction of active connection reported a trembling similarity with the graph of synaptic density reported in Fig.1.3.

Def: *Given* $W(t) \in M_{m,n}(\mathbb{K}), \mathbb{K} \subset \mathbb{R}$ *the connectivity matrix of a network (if not bipartite, m=n), then we define* $A_C : \{0,1\} \times \mathbb{I} \to \mathbb{N}$ *as the active connection function such that*

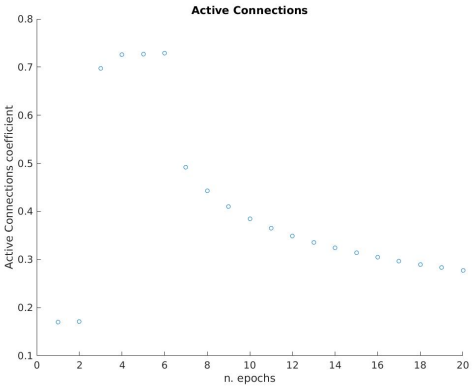
$$A_c(t) = \sum_{i=1}^{m} \sum_{j=1}^{n} \phi_{i,j}(t)$$

$$\phi_{i,j}(t) = \begin{cases} 0 & W_{i,j}(t) = 0 \\ 1 & W_{i,j}(t) \neq 0 \end{cases}$$

$$t \in [t_0, +\infty]$$



(a) Fraction of active connections for H1-H2



(b) Fraction of active connections for H2-H3

Figure 4.2

Indeed, we have that

$$\begin{aligned} A_c(t) &= \sum_{j=1}^{n} d_j(t) \\ &= \sum_{j=1}^{n} d_j(t-1) - [d_j(t-1) \cdot \alpha(s_j(t-1))] \end{aligned} \tag{4.3.8}$$

Hence, we have that the difference in the number of active connections between time $t + \Delta t$, for $\Delta t \in (0, 1)$, and t is

$$\begin{aligned} \Delta A_{c_{(t+\Delta t,t)}} &= A_c(t + \Delta t) - A_c(t) \\ &= \sum_{j=1}^{n} d_j(t + \Delta t) - d_j(t) \\ &= \sum_{j=1}^{n} d_j(t) - [d_j(t) \cdot \alpha(s_j(t + \Delta t))] - d_j(t) \\ &= \sum_{j=1}^{n} -[d_j(t) \cdot \alpha(s_j(t + \Delta t))] \end{aligned} \tag{4.3.9}$$

I have considered $s_j(t+\Delta t)$ instead of $s_j(t)$ under the hypothesis that continuously incoming signals finally determine the discrete update for each t. Thus, for $\Delta t \to 0$

$$\begin{aligned}\frac{dA_c}{dt}(t) &= \frac{d}{dt}(\sum_{j=1}^{n} -[d_j(t)\cdot\alpha(s_j(t))]\,) \\ &= \sum_{j=1}^{n}\frac{d}{dt}(\,-[d_j(t)\cdot\alpha(s_j(t))]\,) \\ &= -\sum_{j=1}^{n}(d_j'(t)\cdot\alpha(s_j(t)) + d_j(t)\cdot\alpha'(s_j(t))\cdot s_j'(t))\end{aligned}$$

Since $d_j(t) = k \in \mathbb{N}$ and the infinitesimal increment can be considered negligible, due to the fact that we are dealing with discrete entities, then $d_j'(t) \equiv 0$, and therefore we have that

$$\begin{aligned}\frac{dA_c}{dt}(t) &= -\sum_{j=1}^{n} d_j(t)\cdot\alpha'(s_j(t))\cdot s_j'(t) \\ &= -\sum_{j=1}^{n}\lambda\cdot\alpha'(s_j(t))\cdot s_j'(t) \qquad (4.3.10)\end{aligned}$$

by considering $d_j(t)$ as a simple scalar that does not significantly impact the magnitude of the increment (which would be the case, as seen later in the chapter). We have therefore established that the infinitesimal increment for the function $A_c(t)$ depends only on the afferring strength to each unit j. From a close analysis of the graphs obtained from trials with ColumnarNet on the MNIST dataset we infer that it exists $t_\epsilon \in [t_0, +\infty]$ such that

$$\frac{dA_c}{dt}(t) = \begin{cases} k \in \mathbb{R}^+ & if\ \ t < t_\epsilon \\ 0 & if\ \ t = t_\epsilon \\ k \in \mathbb{R}^- & if\ \ t > t_\epsilon \end{cases}$$

with t_ϵ characterizing an unstable equilibrium for the function

$$\Rightarrow \frac{d^2 A_c}{dt^2}(t_\epsilon) < 0$$

However, the function $\alpha(s_j(t))$ defined for ColumnarNet provides a realistic account of the dynamics of Fig.4.2a and Fig.4.2b only up to about t=6. Therefore, we now wish to find a better function, coherent with experimental data.

The problem has then circulated back to the analysis of the strength afferring to each unit at each epoch. In order to gain some insight into the global behaviour of the system, I have considered suitable to define the total strength of the connectivity matrix as

$$s(t) = \sum_{j=1}^{m} |s_j(t)| \tag{4.3.11}$$

in order to avoid fluctuations due to the presence and, or prevalence, of both inhibitory and excitatory connections. The resulting plot (see Fig.4.3) is, averaged for the two RBMs in order to compensate for differences in their size

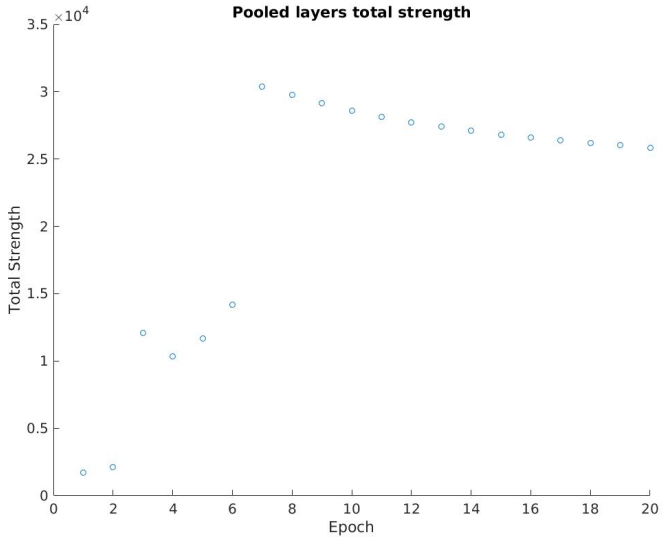


Figure 4.3: Total strength as a function of epochs

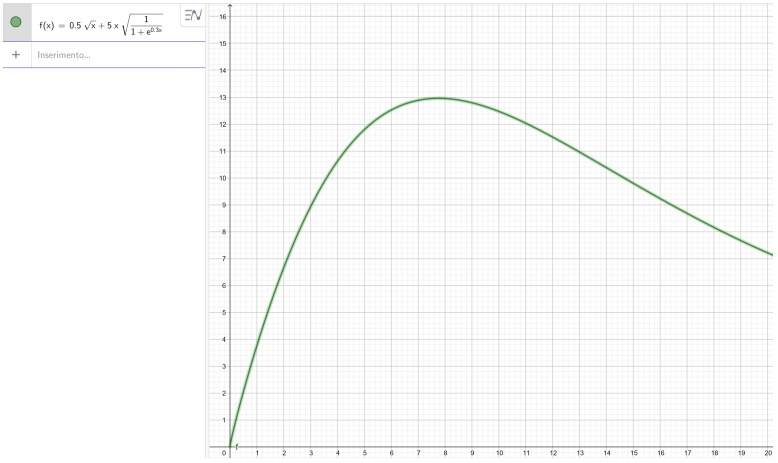


Figure 4.4: s(t) rough approximation with GeoGebra software

After a conspicuous number of trials with the open software "GeoGebra", it was possible to approximate (very roughly) the curve interpolating points in Fig. 4.3 as (Fig.4.4)

$$s(t) = \delta\sqrt{t} + \gamma t\sqrt{\frac{1}{1+e^{\beta t}}} \qquad \delta, \gamma, \beta \in \mathbb{R} \tag{4.3.12}$$

By further elaborating on $s(t)$ with the tools of differential calculus, we have that

$$\begin{aligned}\frac{ds}{dt}(t) &= \frac{\delta}{2\sqrt{t}} + \gamma\left(\sqrt{\frac{1}{1+e^{\beta t}}} + \frac{t}{2}\sqrt{1+e^{\beta t}}\left(-\frac{1}{(1+e^{\beta t})^2}\right)\beta e^{\beta t}\right) \\ &= \frac{\delta}{2\sqrt{t}} + \frac{\gamma}{\sqrt{1+e^{\beta t}}}\left(1 - \frac{\beta}{2}\frac{te^{\beta t}}{(1+e^{\beta t})}\right)\end{aligned} \tag{4.3.13}$$

Let us therefore suppose that, on a well defined network where the strengths directly depend on the weights of the connection matrix, and consequently on the number of active connections, it is possible to parameterize the strength in terms of the time variable t. A new possible

update rule could be

$$
\begin{aligned}
d_j(t+1) &= d_j(t) + \lambda |s_j(t)| \frac{ds}{dt}(t) \\
&= d_j(t) + \lambda |s_j(t)| \left(\frac{\delta}{2\sqrt{t}} + \frac{\gamma}{\sqrt{1+e^{\beta t}}} \left(1 - \frac{\beta}{2} \frac{te^{\beta t}}{(1+e^{\beta t})} \right) \right)
\end{aligned} \tag{4.3.14}
$$

Drawing from 4.3.10, we can see how $\alpha(s_j(t)) = - \mid s_j(t) \mid \frac{ds}{dt}(t)$ and $d_j(t) = \lambda$ by supposing an approximately equal strength differential $\forall j \in h_{i+1}$. Consequently

$$
\begin{aligned}
\frac{dA_c}{dt}(t) &= \frac{d}{dt} \left(\sum_{j=1}^{n} \lambda |s_j(t)| \frac{ds}{dt}(t) \right) \\
&= \sum_{j=1}^{n} \frac{d}{dt} \left(\lambda |s_j(t)| \frac{ds}{dt}(t) \right) \\
&= \lambda \sum_{j=1}^{n} \left((\frac{d}{dt} |s_j(t)|) \frac{ds}{dt}(t) + |s_j(t)| (\frac{d}{dt} \frac{ds}{dt}(t)) \right) \\
&= \lambda \left(\frac{ds}{dt}(t) \sum_{j=1}^{n} (\frac{d}{dt} |s_j(t)|) + \frac{d^2 s}{dt^2}(t) \sum_{j=1}^{n} |s_j(t)| \right) \\
&= \lambda \left(\frac{ds}{dt}(t) \frac{d}{dt} \left(\sum_{j=1}^{n} |s_j(t)| \right) + s(t) \frac{d^2 s}{dt^2}(t) \right) \\
&= \lambda \left(\left(\frac{ds}{dt}(t) \right)^2 + s(t) \frac{d^2 s}{dt^2}(t) \right)
\end{aligned} \tag{4.3.15}
$$

Thus, I was finally able to make the infinitesimal increment in the number of active connections in the entire graph depends exclusively on the total strength of the connectivity matrix, hence obtaining a global descriptor of the network's dynamics. Finally, connections were either preserved or discarded based solely on their absolute value, those with the highest surviving and the other 'perishing'.

5

Methods

The experiments, entirely carried out in the MATLAB ecosystem, can be effectively subdivided in sequential phases, each of which maintained, unvaried, a canonical DBN architecture taking as input a real valued vector $v \in \mathbb{R}^{784}$ from the MNIST dataset as control condition. The MNIST dataset [14] contains 60000 training and 10000 test examples of written digits ranging from 0 to 9. The DBNs all have hidden layers of dimensions 400, 400, and 2000 and have all been trained with the Contrastive Divergence algorithm [8], with momentum and sparsity constraints imposed on the third layer. The resulting weight matrices were used to create the relative adjacency matrices; formally, given $W_{h_i,h_{i+1}}$ the weight matrix of the connections between h_i and h_{i+1} at the end of the training process, the relative adjacency matrix is

$$Adj_{h_i,h_{i+1}} = \begin{pmatrix} 0_{|h_i|,|h_i|} & W_{h_i,h_{i+1}} \\ W^T_{h_i,h_{i+1}} & 0_{|h_{i+1}|,|h_{i+1}|} \end{pmatrix}$$

Each of the trained models was later used for testing on a Perceptron's recognition task [22][25]*.

5.1 Dynamic Learning

The first relevant condition consists in the implementation of the algorithm described in section 4.3.2, following the modification of the canonical DBN algorithm. Its architecture is analogous to that of the control model, therefore having hidden layers of dimensions 400, 400, and 2000, respectively. The weights between the input and first hidden layer h_1 are initialized in a classical fashion, with all the coefficients of the weight matrix taking values according to a normal distribution. Instead, the weight matrices between the first and second, and second and third hidden layers are initialized as $W_f(0)$, with the central blocks $W_{n_k,m_k}(0)$ expressing full connectivity and the blocks Φ_{n_s,m_t}, for $s \neq t$, having coefficients different from zero if and only if $\mid \Phi_{n_s,m_t i,j} \mid > 0.15$. Furthermore, the parameters in 4.3.12 and 4.3.14 have been set to $\delta = 0.5$, $\gamma = 5$, and $\beta = 0.3$.

The training phase is characterized by the creation of a Struct containing the tensors of the weight matrices for each epoch of each layer. Such Struct has later been used for the computation and plotting of relevant measures such as fraction of active connections at each time step and total absolute strength at each time step. Furthermore, I have exploited relevant statistical graph measures such as Degree Centrality, Eigen Centrality, and Betweenness Centrality, computed by means of the algorithms and functions provided by the MATLAB environment. Most of this measures have been computed, instead than on the adjacency matrices themselves, on their unipartite (upward) projections, both weighted and unweighted. In order to effectively project the bipartite structure identified by the weight matrix, I relied on the algorithm furnished by Banerjee, Jenamani, and Pratihar [2], with due modifications for the weighted version. I opted for the projection of such structures mainly to dodge the unfeasibility of computing the (weighted-) clustering coefficient on the adjacency matrices, which would trivially return 0 for each node. Furthermore, the unipartite projection of the

*The complete source code is available at http://ccnl.psy.unipd.it/research/deeplearning

connectivity matrices may provide a mean to infer the latent intralayer connectivity of each hidden layer, which is deliberately omitted in RBMs.
Finally, I experimented with a simpler version of *Dynamic Learning II*, with

$$\frac{ds}{dt}(t) = \begin{cases} 0.8 & if \quad t < 7 \\ -1.2 & else \end{cases}$$

to assess whether the peculiar performances obtained have a tight correlation with the function 4.3.13 or whether any function leading to that fraction of connections generates a desirable topology and related optimal functionality; further investigation in this direction are encouraged.

5.2 Noise Trials

The assessment of a similarity in terms of performances between the canonical and modular models, and the observation of substantial topological differences in the connectivity between $H1 - H2$ and $H2 - H3$, has naturally led me, thanks to the suggestions of Dr. Testolin, to experiment in altered conditions. The main aim was to test whether the topological differences correlates with a difference in terms of performances when the available data only resembles, to different extents, the data distribution that needs to be learned.

5.2.1 Noise in the training dataset

In the first experiment, models have been trained both in a canonical way and according to the modular algorithm in under-optimal conditions. The circumscribed aim of this first procedure was to assess whether given distorted examples to the model, it would nonetheless prove able to extract the features most relevant for the Perceptron's recognition task. Specifically, ten different models for each algorithm were trained on a training dataset provided with Gaussian noise with mean $\mu = 0$ and variance $var = 0.1$ and tested on a unvaried test

dataset. Later assessment of statistical differences between the performances of the two algorithms has been conducted by means of a one-tail t-test for both $\alpha = 0.05$ and $\alpha = 0.01$, on the representations generated by the third hidden layer.

5.2.2 Noise in the test dataset

A complementary approach envisions the testing of both canonical and modular models trained on an incorrupt training dataset on testing datasets corrupted by increasing levels of noise; with this aim in mind, both models have been exposed to test datasets with sequentially increasing Gaussian noise ($\mu = 0, Var = 0.3 \cdot k, \;\; k \in \{1, .., 10\}$). The specific goal of this procedure is to establish whether an evolving network topology as the one detailed in 4.3.2 is more resilient to noise in the data and, if so, to provide a valuable starting point to investigate the properties that grant such an advantage.

6

Results

The first step in establishing whether the architecture outlined in *Dinamic Learning II* could actually be of some, if any, impact to future research has been to verify the actual accuracy of the model on the original MNIST dataset. If the model had proved effective enough to reach level of accuracy that approximated those of a standard architecture within 0.02 decimal points, then it would have been considered worth of further investigation. Luckily enough, the model has not only been able to reach level of accuracy within the established interval, but it has consistently outperformed the standard one in both test and training accuracy of $0.00\mathrm{E}$, for $\mathrm{E} \in \mathbb{R}$. These first, marginal results have been sufficient to spur further interest in a more formal analysis of the analogies and differences between the two architecturally dissimilar models.

6.1 Accuracy, Active Connections, and Strength

I begin the analysis from a qualitative assessment of the differences of the receptive fields of neurons in hidden layers 1^{st}, 2^{nd}, and 3^{rd} between the standard and modular architectures. Because the 1^{st} hidden layer receives its input directly from the *'Visual Field'*, it would be inadvisable to enforce a modular topology on the connections; consequently, the receptive fields of neurons in $H1$ (Fig.6.1) are the same for both models.

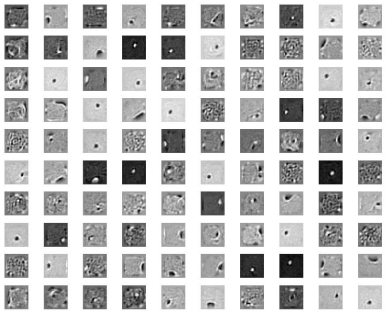
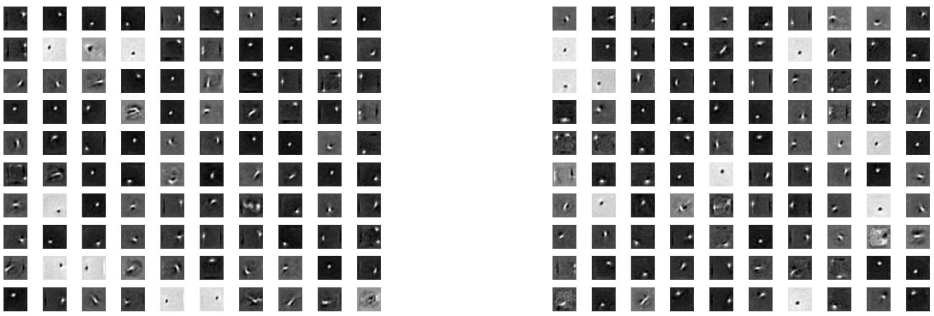


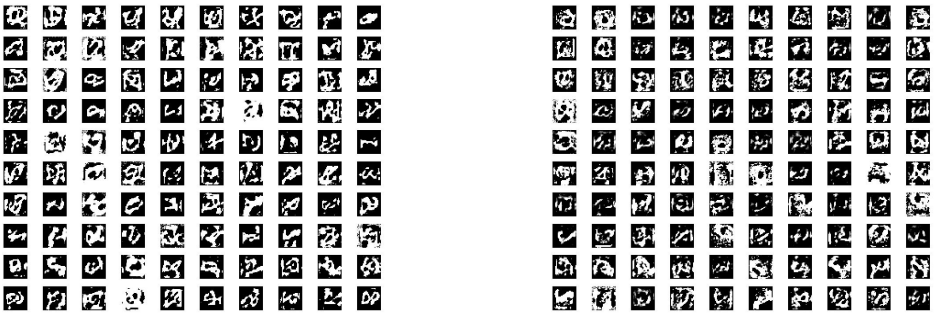
Figure 6.1: Receptive Fields for Neurons in H1 for both the standard and modular architecture

However, $H2$ (see Fig.6.2) and $H3$ (see Fig.6.3) should actually differ in terms of receptive fields, as the topologies of the connections innervating their units present, as we will later see, substantial differences in terms of active (and therefore existing) edges.



(a) Receptive field for neurons in H2 in the standard DBN **(b)** Receptive field for neurons in H2 in the modular DBN

Figure 6.2



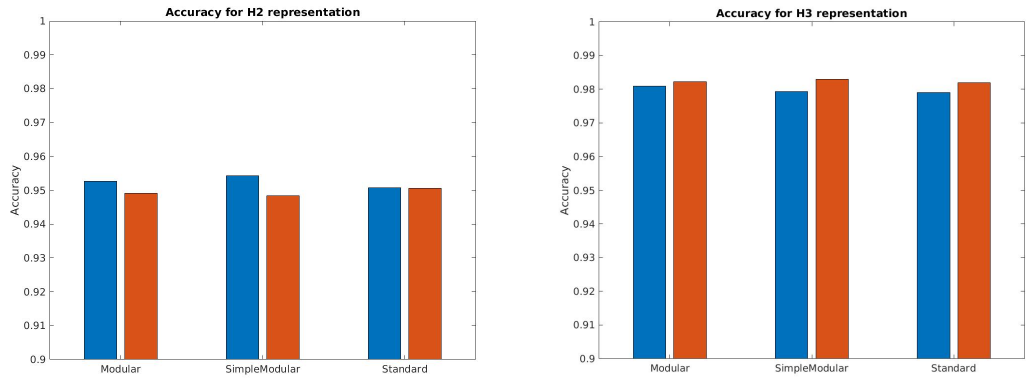
(a) Receptive field for neurons in H3 in the standard DBN **(b)** Receptive field for neurons in H3 in the modular DBN

Figure 6.3

As can be readily observed from Fig.6.2 and 6.3, the major qualitative difference between the receptive fields generated by the two models is the *'resolution'* of those fields, with those belonging to the standard architecture having higher resolution due to the greater number of active connections propagating information forward. Nonetheless, I acknowledge that such differences are indeed minimal, and this could partially explain the relative stability in accuracy scores across architectures. Specifically, the accuracy reached on a recognition task by the Perceptron using the representations generated by $H2$ and $H3$ are

- Standard DBN
  $H2$:
  Test Mean = 0.9543
  Test Std = $8.75e-05$
  Train Mean = 0.9506
  Train Std = $8.89e-05$
  $H3$:
  Test Mean = 0.9791
  Test Std = $4.71e-05$
  Train Mean = 0.9819
  Train Std = $4.79e-05$
- Modular DBN
  $H2$:
  Test Mean = 0.9526
  Test Std = $6.99e-05$
  Train Mean = 0.9491
  Train Std = $7.33e-05$
  $H3$:
  Test Mean = 0.9810
  Test Std = $1.56e-04$
  Train Mean = 0.9823
  Train Std = $6.77e-05$
- Simple Modular
  $H2$:
  Test Mean = 0.9507
  Test Std = $5.16e-05$
  Train Mean = 0.9484
  Train Std = $1.49e-04$
  $H3$:
  Test Mean = 0.9793
  Test Std = $8.75e-05$
  Train Mean = 0.9830
  Train Std = $6.71e-05$

as clearly observable from Fig.6.4. As a result in itself it is pretty outstanding if we consider the number of active connections at the end of the training process, with a connection being



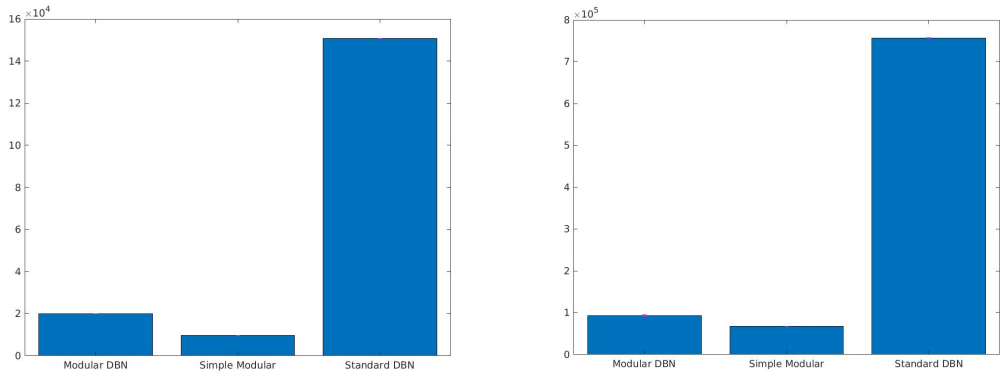
(a) Accuracy Scores for representations generated by H2 **(b)** Accuracy Scores for representations generated by H3

Figure 6.4: Training - Red; Testing - Blue

active if it has a weight different from 0 and inactive otherwise. I have also further imposed, on the canonical architecture, a threshold to reduce the number of active connections, such that

$$w_{i,j} \begin{cases} active & if \mid w_{i,j} \mid > 0.01 \\ inactive & otherwise \end{cases}$$

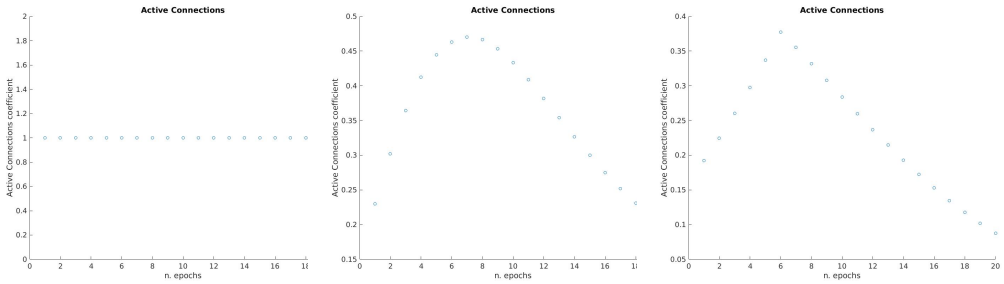
Indeed, as highlighted by the bar graphs, the number of active connections in the modular architecture is $\eta(n \cdot m)$, with $\eta \in [0.05, 0.2]$ and $n \cdot m$ the approximate number of active connections (see Fig.6.5) in the standard one. Both graphs 6.4 and 6.5 are comprehensive of error bars, which are however undetectable due to their relative small size (smaller of three orders of magnitude) if compared to the bars themselves. It is therefore useful to turn again to a qualitative analysis, this time of the fraction of the number of active connections per epoch and the total strength $s(t)$ (4.3.11) at each epoch, to assess whether the trajectories inferred from such graphs are coherent with what was theorized in 4.3.2. Since the rules governing connectivity between any two layers except the input and 1^{st} hidden layer are the same, I will proceed with the analysis of such properties for the connectivity of $H2 - H3$ only.
As could be expected from Fig.6.5 and from the initial phase of *'synaptic burst'* characterizing the first epochs, the scatter-plots for 4.3.7 follow very specific trajectories, as displayed in



(a) Number of active connections between H1 and H2

(b) Number of active connections between H2 and H3

Figure 6.5



(a) Fractions of active connections between H2 and H3 (Standard)

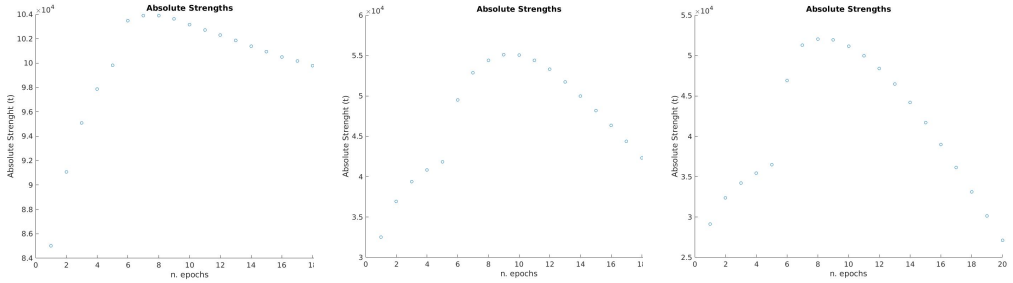
(b) Fraction of active connections between H2 and H3 (Modular)

(c) Fraction of active connections between H2 and H3 (Simple M.)

Figure 6.6

Fig.6.6. for the standard, modular, and simple modular architecture, respectively. As outlined by the three plots, Fig.6.6b and 6.6c display a temporal evolution of the neural structure, whereas Fig.6.6a portrays a static fully connected network. Furthermore, the complete modular architecture has obtained better performances on the test dataset if compared to the simple one, and it is also the one with the smoother dynamical evolution of the network topology. However, it must be noted that the simple modular architecture has less than half of the active connection of the complete modular one; consequently, further tests are encouraged to determine differences between the two models.

Finally, we turn to the analysis of the total strength at each epoch, to verify if the hypothesized



(a) Total Strength in H2-H3 (Standard)

(b) Total Strength in H2-H3 (Modular)

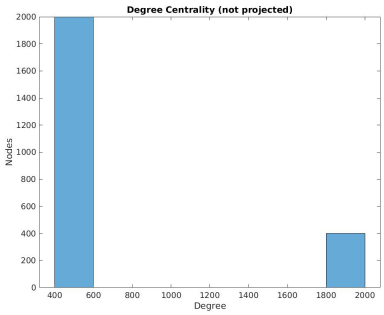
(c) Total Strength in H2-H3 (Simple M.)

Figure 6.7

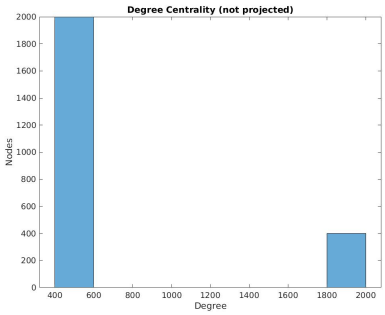
approximation in 4.3.12 could indeed be used as a starting point to formalize a more rigorous and harmonious model of the dynamical evolution of a neural network. It could be useful to search for functions that describe particular properties of the network in a global manner rather than focusing on its single units, much as state functions allow to describe ideal thermodynamics systems in terms of collective rather than individual particles behaviour.
The graphs for the total strength are shown in Fig.6.7. At first inspection, it is unequivocally observable how Fig.6.7a and 6.7b propose, to different extents, valid approximations of 4.3.12; instead, the trajectory in Fig.6.7c is far too different from the comparison, albeit this could be primarily due to the values arbitrarily chosen for $\frac{ds}{dt}(t)$. Thereby, it may be possible to hypothesize that, in order to create appropriate representations of the input, the total strength of the neural network (at the microscopic level) needs to follow, if considered along a time continuum, a trajectory resembling the one provided by 4.3.12.

6.2 Degree Centrality, Betweenness Centrality, and Clustering Coefficient

Given the previous results, it seems spontaneous to ask whether such invariance in terms of performance spurs from peculiar properties of the network, and whether such properties directly emerge from the dynamic evolution of the network topology. In order to answer to this question I relied on statistical measures of graphs, specifically those outlined in chapter

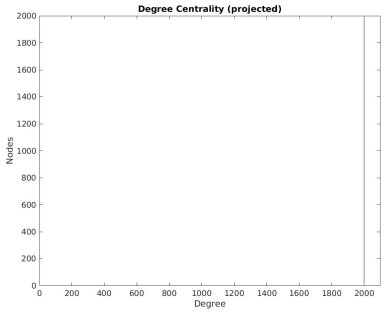


(a) Degree Centrality for H2-H3 in the un-projected Network at t=0

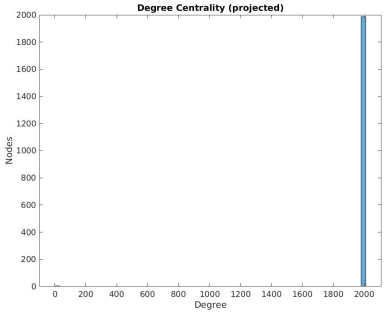


(b) Degree Centrality for H2-H3 in the un-projected Network at t=20

Figure 6.8



(a) Degree Centrality for H3 in the projected Network at t=0

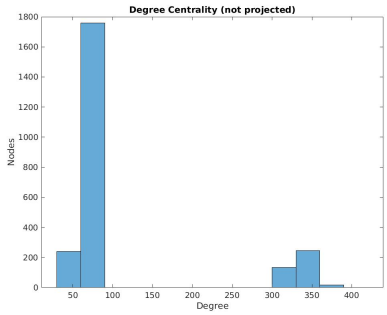


(b) Degree Centrality for H3 in the projected Network at t=20

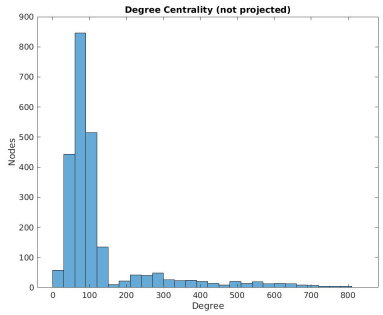
Figure 6.9

2; an equivalently interesting analysis, given the time dependent network topology, would be the investigation of the architecture through persistent homology, although this is far beyond the purpose of this thesis.

In the standard DBN the measure of degree centrality remains the same throughout the whole training process, as could be expected from the fact that the weight matrix identifies a fully connected network (see Fig.6.8 and 6.9), i.e. all nodes have maximum connectivity. Instead, it is possible to observe a clear shift in the relevance of the nodes in the network by virtue of their degree centrality score for the modular DBN. Indeed, for the projected scores (see Fig.6.11), it appears that the training process results in an overall decrease in the degree

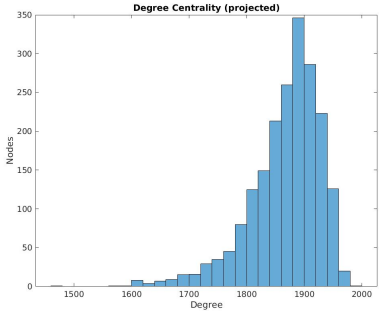


(a) Degree Centrality for H2-H3 in the un-projected Network at t=0

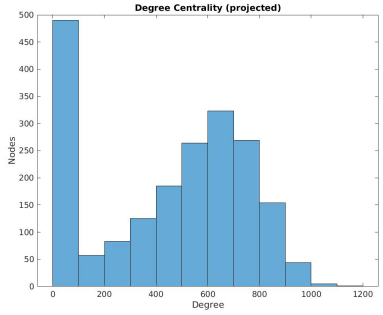


(b) Degree Centrality for H2-H3 in the un-projected Network at t=20

Figure 6.10



(a) Degree Centrality for H3 in the projected Network at t=0

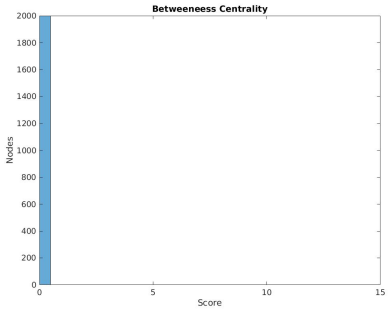


(b) Degree Centrality for H3 in the projected Network at t=20

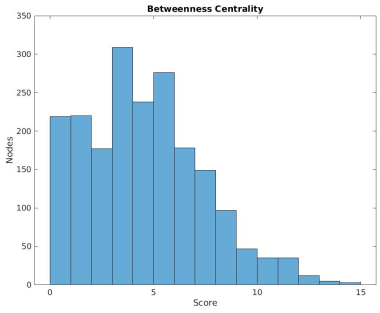
Figure 6.11

of each node in the network, which is consistent with the need to maximize the efficiency of intralyer information processing. Instead, the unprojected scores (see Fig.6.10) reveal an increase in the degree of each node; this difference in trend may be due to the way in which a unipartite projection is defined. However, from 6.10 it is clear how at the end of the training process there will be a relatively small number of nodes with very high degree; this small subset could comprehend the central hub of the network, through which most of the relevant information flows. I therefore turn to the assessment of other centrality measures that may furnish an explanation to the hubs hypothesis; for the present purpose, I have chosen the Eigen Centrality and the Betweenness Centrality measures.

Since results on the Eigen measure have been largely irrelevant to the stated purpose, it will not be reported; nonetheless, the reader is encouraged to experiment with it, as well as with other centrality measures, in order to evaluate other important network properties.
As stated in chapter 5, scores for Betweenness Centrality (see Fig.6.12 and 6.13) have been computed solely on the upward unipartite projection of $H2 - H3$, thereby on a possible latent connectivity among the units in $H3$.

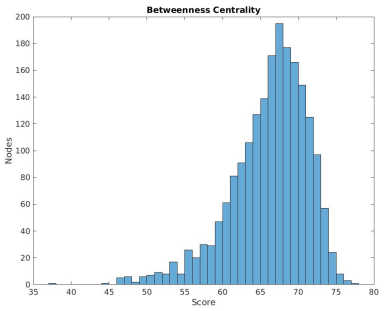


(a) Betweenness Centrality for H3 at t=0

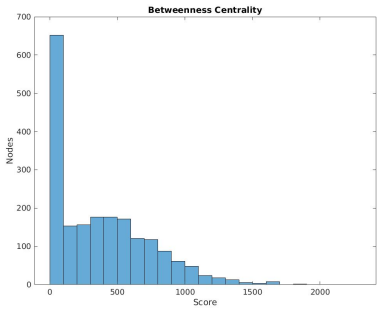


(b) Betweenness Centrality for H3 at t=20

Figure 6.12: Standard DBN



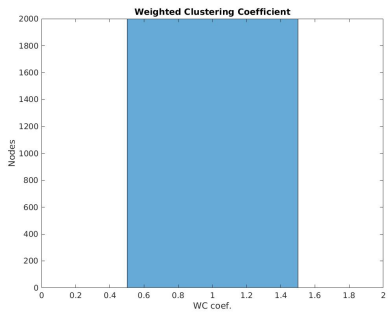
(a) Betweenness Centrality for H3 at t=0



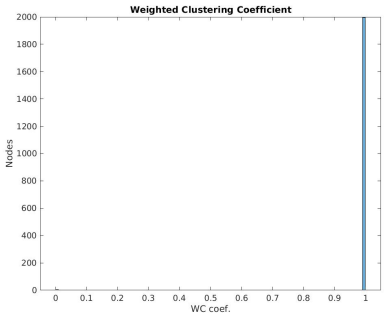
(b) Betweenness Centrality for H3 at t=20

Figure 6.13: Modular DBN

Of particular interest are the results for the modular architecture (see Fig.6.13), where the Betweenness score for a small subset of nodes increases of two orders of magnitude. This means that for most nodes in the network, and especially for those in that small subset, the number of geodesics between two generic nodes s and t travelling through a third, different



(a) Weighted clustering coefficient at t=0

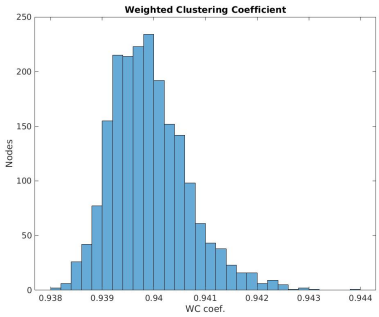


(b) Weighted clustering coefficient at t=20

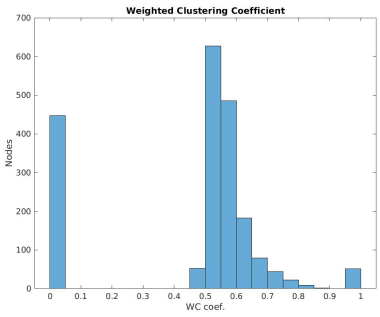
Figure 6.14: Standard DBN

node r belonging to the subset dramatically increases, thus prompting to suggest that the network evolves its topology so to maximize the efficiency of information processing among units in the same layer. Moreover, if the nodes with the highest Betweenness score also happen to be those with the highest degree, then this would provide a valuable contribution to the hubs hypothesis. Indeed, by using the temporal evolution of the modular architecture as a paradigm for an ideal simple biological neuronal network, it could be argued that the process of *synaptic burst* and *synaptic pruning* efficiently organize the cortical architecture as to minimize paths length among neurons that collectively code for a specific type of information.

Finally, I report on the weighted clustering coefficient for the upward unipartite $H2 - H3$ projection at $t = 0$ and $t = 20$, for both architectures. Again, as expected in the standard architecture (see Fig.6.14) the weighted clustering coefficient is, for every node in the projected network, equal or very close to one, since each node has degree 1999 and each of the edges has weight of approximately 2000. Instead, in the modular architecture (see Fig.6.15) the weighted clustering coefficient at $t = 0$ seems to be, approximately, normally distributed around the mean 0.94. As evincible from 6.15b, the training process elides many of the $1 - loops$, or triangles, present at $t = 0$, and therefore suggests that the latent connectivity of $H3$ is much *'lighter'*, hence with far less intralayer connections. From a qualitative inspection of 6.14b and 6.15b I can, perhaps boldly, infer that the dynamic evolution



(a) Weighted clustering coefficient at t=0



(b) Weighted clustering coefficient at t=20

Figure 6.15: Modular DBN

of the modular architecture leads to a final latent intralayer architecture way more compatible with available biological data than its standard counterpart. Furthermore, with regard to the question raised at the beginning of the section, it may be claimed that the observed invariance allegedly stems from a much more efficient organization of the network, which proves fundamental in preserving performances despite the drastically inferior number of active connections; however, we still need a formal proof for this hypothesis.

6.3 Noise Trials

Invariance of performances despite the significant architectural differences has prompted to a more in-depth study under altered conditions, i.e. conditions in which either the training or test dataset are compromised to different extents.

6.3.1 Training

In the first altered condition, detailed in 5.2.1, the modular architecture has proven superior to the standard one, in terms of performances reached on the representations generated by $H3$, as demonstrated by the one-tail t-test applied to both training and test scores. Specifically, my null hypothesis is that there is no relevant statistical difference between the performances obtained by the two different architectures, whereas the alternative hypothesis states that

there is indeed a difference between the two. The obtained results are the following

- Training Accuracy:
  Mean (Standard DBN) = 0.87906
  Std (Standard DBN) = $4.55e-03$
  Mean (Modular DBN) = 0.89406
  Std (Modular DBN) = $4.53e-03$
  p-value = $3.80453874055226E-07$

- Test Accuracy:
  Mean (Standard DBN) = 0.89011
  Std (Standard DBN) = $4.72e-03$
  Mean (Modular DBN) = 0.89994
  Std (Modular DBN) = $5.05e-03$
  p-value = 0.000141257646862

Hence, I can safely dismiss the null hypothesis (both for $\alpha = 0.05$ and $\alpha = 0.01$) and state that there is indeed a statistically significant difference between the accuracy of the two architectures if trained on the same degraded dataset. From this piece of information I can conjecture that the modular model is better at retrieving the most salient features of the dataset even in the presence of training noise; thus, the properties of the architecture could be subject of further study in order to improve the performance of the most deployed neural network's models in situations where datasets tend to incorporate substantial noise.

6.3.2 Testing

In the second altered condition, detailed in 5.2.2, results have not been as satisfying as those in 6.3.1, especially for the representations generated by $H2$. Nonetheless, from this first analysis it appears that higher order representations generated by a dynamic modular architecture on a canonical training dataset encapsulate features that are more relevant for the recognition task of noisy, not previously encountered data. The main difference between the preceding and present point is that in the former the model proves superior at extrapolating features from noisy input, whereas in the second at using extracted features to classify noisy data.

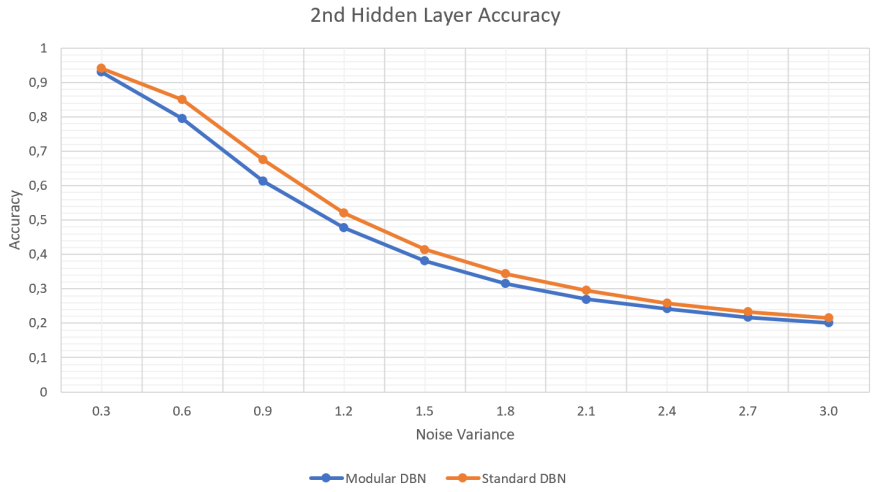


Figure 6.16: Performance on representations created by H2 generated from increasingly noisy datasets

From the inspection of Fig.6.16 it is evident how the standard architecture outperformed the modular one in the recognition task on the representation generated by the second hidden layer. This result could be due to the fact that the second hidden layer has only 400 units, and therefore a full connectivity between that and the layer below proves marginally better at encoding relevant information. Indeed, the greatest difference in terms of accuracy is registered for small noise variance, where greater encoding potential actually reflects a major capability to exploit collective information distributed throughout the network. However, as the noise variance increases the difference becomes negligible, thus proving the *'breakdown'* of both architectures. Accordingly to what has been previously seen in Fig.6.16, also for the representations generated by H3 the performance of the standard architecture is better than that of the modular one for small noise variance. This may again be due to the *'higher resolution'* of the representations. However, the models seem to quickly reach a *'tipping point'* (see Fig.6.17) from where the previous trend is inverted, this time permanently. From the instance in which the variance of the noise in the test dataset is $Var = 0.9$, the accuracy reached by the two models begin to diverge, with the better performances registered by the modular architecture. Even more interestingly, and differently from the results for layer $H2$, the difference resulting from such initial divergence seems to remain constant independently from

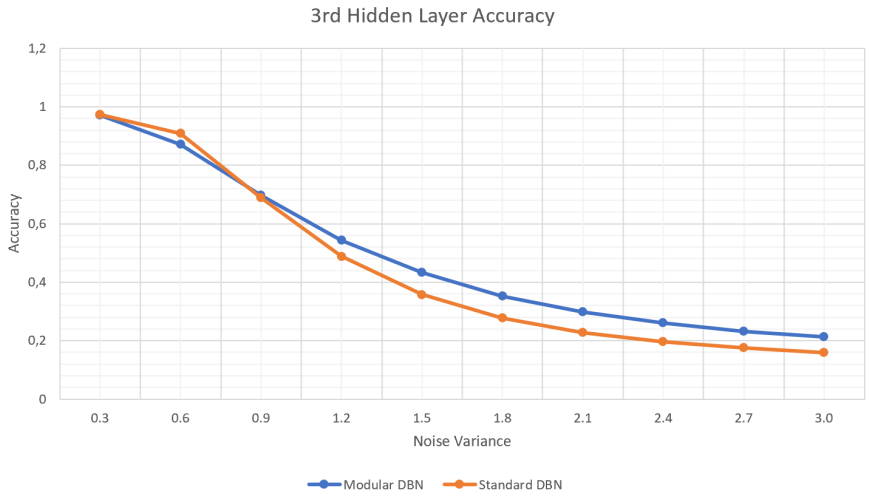


Figure 6.17: Performance on representations created by H3 generated from increasingly noisy datasets

the noise variance, thus suggesting that the modular architecture is indeed better at extrapolating the most relevant feature from the original dataset. The results displayed in Fig.6.16 and Fig.6.17 may be considered statistically significant, as in both cases error bars generated from results of 10 different trials were so small as to be invisible to the human eye.

..It was on Earth that the positronic brain was invented and on Earth that robots had first been put to productive use...

Caves of Steel, Asimov

7

Conclusion

Evolution has endowed mankind with the capability to perform very complex computation, resulting in the expression of abstract thought, at a very low energy cost. At the moment, the same cannot be said about Artificial Neural Networks, which tend to be limited in size and very computationally expensive to train and deploy in industrial settings. This is perhaps partly due to an inefficient implementation of the connectivity among neurons, or nodes of the representing graph, which almost always instantiates fully connected topologies; other than being biologically implausible, it seems to be also unnecessary, as demonstrated in the present thesis.

Indeed, I have shown how an appropriate pruning process of the ANNs' connectivity through the dynamic evolution of their topology yields equally accurate performances with a dramatically inferior number of active (and therefore existing) connections. Furthermore, the process of dynamic evolution has also revealed important emergent network properties, such as the drastic increase in Betweenness Centrality score for a small subset of nodes as well as the

soaring in the degree of nodes in an equally small subset. It is thus possible to envision the time-dependent formation of hubs governing the microscopic behaviour and information processing of small network structures.

Future research may address the generalization of such procedure to other neural network structures capable of processing information from more complex datasets, as well as the creation of algorithms aimed at the exploitation of the incredibly high sparsity of the connectivity matrices to sensibly reduce the testing time, and therefore aid in the process of industrial adoption of A.I. systems. Moreover, the proposed rudimentary analysis may serve as a valuable starting point to theorize a more rigorous model of collective neural networks' dynamics, thereby promoting *'vertical'* advances in a field where most innovation is of a *'horizontal'* nature, meaning that it mainly proposes functional amelioration to networks through modification of existing paradigms. Instead, the development of a rigorous and comprehensive framework must be sought in order to significantly advance the field.

As a final note, this thesis is meant to be a, albeit very simple, manifesto for the endless possibilities and potential of highly formal research in the field of Neuroscience, which may catalyze a relatively near technological revolution.

Bibliography

[1] Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive science*, 9(1), 147–169.

[2] Banerjee S., Jenamani M., P. D. (2017). Algorithms for projecting a bipartite network. In *2017 Tenth International Conference on Contemporary Computing (IC3)*.

[3] Barrat, A., Barthelemy, M., Pastor-Satorras, R., & Vespignani, A. (2004). The architecture of complex weighted networks. *Proceedings of the national academy of sciences*, 101(11), 3747–3752.

[4] Bishop, M. J. (2015). History and philosophy of neural networks. *Computational Intelligence*.

[5] Edelsbrunner, H. & Harer, J. (2010). *Computational topology: an introduction*. American Mathematical Soc.

[6] Fletcher, J. M. & Wennekers, T. (2018). From structure to activity: using centrality measures to predict neuronal activity. *International journal of neural systems*, 28(02), 1750013.

[7] Freberg, L. (2009). *Discovering biological psychology*. Cengage Learning.

[8] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8), 1771–1800.

[9] Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7), 1527–1554.

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

[11] Huttenlocher, P. R. (2002). *Neural plasticity: The effects of environment on the development of the cerebral cortex*. Harvard University Press.

[12] Kenny, P. (1988). Notes on boltzmann machines. *https://www.crim.ca/perso/patrick.kenny/BMNotes.pdf*.

[13] Koller, D. & Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

[14] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.

[15] Minsky, M. & Papert, S. (1969). Perceptron: an introduction to computational geometry. *The MIT Press, Cambridge, expanded edition*, 19(88), 2.

[16] Newman, M. E. (2003). The structure and function of complex networks. *SIAM review*, 45(2), 167–256.

[17] Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. (1988). Learning representations by back-propagating errors. *Cognitive modeling*, 5(3), 1.

[18] Salakhutdinov, R. & Hinton, G. (2009). Deep boltzmann machines. In *Artificial intelligence and statistics* (pp. 448–455).

[19] Smolensky, P. (1986). *Information processing in dynamical systems: Foundations of harmony theory*. Technical report, Colorado Univ at Boulder Dept of Computer Science.

[20] Sutskever, I. & Tieleman, T. (2010). On the convergence properties of contrastive divergence. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 789–795).

[21] Testolin, A., Piccolini, M., & Suweis, S. (2019). Deep learning systems as complex networks. *Journal of Complex Networks*, 2051-1329.

[22] Testolin A., Stoianov, I. D. F. D. G. M. . Z. M. (2013). Deep unsupervised learning on a desktop pc: A primer for cognitive scientists. *Frontiers in Psychology*, 4, 251.

[23] Tucker, A. (2006). *Applied Combinatorics*. John Wiley & Sons, Inc.

[24] Vilis, T. (2017). *The Physiology of Senses*. Amazon Digital Services LLC.

[25] Zorzi, M., T. A. . S. I. (2013). Modeling language and cognition with deep unsupervised learning: a tutorial overview. *Frontiers in Psychology*, 4, 515.

Acknowledgments

If I am now able to run, it is because I saw those preceding me walking down aisles of uncertainty and novelty. With this premise, I would like, first and foremost, to thank my supervisor Dr. Alberto Testolin, who other than being an excellent supporter during my internship and thesis, has also been, since my first year at the University of Padua, a superb counselor. In the same spirit, I would like to thank my co-supervisor Dr. Samir Simon Suweis, who has, since the beginning of my project, stimulated my thought process through the exploration of unconventional research avenues and, perhaps more important, has continuously supplied me with opportunities to expand my knowledge of the field of Computational and Theoretical Neuroscience. More broadly, I would like to thank the University of Padua for providing ample opportunities to navigate interdisciplinary subjects in the best way possible, that is with seminars, lectures and workshops on the present and related topics.

I would also like to thank my close and extended family, who has proved vital in providing the resilience necessary to tackle the encountered difficulties. Similarly, I am grateful to family friends for the continuous support. Last, but not least important, I would like to thank my friends for having shared part of this journey with me and, above all, for having tolerated my ideals of adventure.

Finally, and perhaps egoistically, I would like to thank that part of myself that has refused to give up this unconventional journey, despite the personal sacrifices, when all seemed to have very little sense.

I finally glimpse a new Dawn.