Deep Neural Networks via Complex Network Theory: A Perspective

Emanuele La Malfa[1,] , **Gabriele La Malfa**[2] , **Giuseppe Nicosia**[3] and **Vito Latora**[3,4]
[1]University of Oxford
[2]King's College London
[3]University of Catania
[4]Queen Mary University of London
{emanuele.lamalfa@cs.ox.ac.uk gabriele.la_malfa@kcl.ac.uk giuseppe.nicosia@unict.it
v.latora@qmul.ac.uk}

Abstract

Deep Neural Networks (DNNs) can be represented as graphs whose links and vertices iteratively process data and solve tasks sub-optimally. Complex Network Theory (CNT), merging statistical physics with graph theory, provides a method for interpreting neural networks by analysing their weights and neuron structures. However, classic works adapt CNT metrics that only permit a topological analysis as they do not account for the effect of the input data. In addition, CNT metrics have been applied to a limited range of architectures, mainly including Fully Connected neural networks. In this work, we extend the existing CNT metrics with measures that sample from the DNNs' training distribution, shifting from a purely topological analysis to one that connects with the interpretability of deep learning. For the novel metrics, in addition to the existing ones, we provide a mathematical formalisation for Fully Connected, AutoEncoder, Convolutional and Recurrent neural networks, of which we vary the activation functions and the number of hidden layers. We show that these metrics differentiate DNNs based on the architecture, the number of hidden layers, and the activation function. Our contribution provides a method rooted in physics for interpreting DNNs that offers insights beyond the traditional input-output relationship and the CNT topological analysis.

1 Introduction

Deep Neural Networks (DNNs) are learning algorithms loosely inspired by the human brain: they consist of layers of interconnected nodes called *neurons* that process an input to produce an output [Bishop, 1995; Schmidhuber, 2015; Goodfellow *et al.*, 2016]. Such models perform remarkably on many tasks without requiring humans to engineer features manually, as each DNN layer delegates neurons to learning and representing specific features, the so-called *hierarchical representation* of the input [Singh *et al.*, 2018]. However, the outstanding ability of DNNs to learn these representations is accompanied by the challenge of *interpreting* both what happens inside the neural networks and the mapping process between a representation (data) and its label (output) [Montavon *et al.*, 2018; Ghorbani *et al.*, 2019]. Since the onset of the modern era of machine learning, interpreting the learning mechanisms of neural networks has emerged as a primary area of research, as highlighted in seminal works such as Erhan et al., Karpathy, and Zeiler et al. [2009; 2015; 2014]. In this work, we seek to unify, understand and represent DNNs and their dynamics through the lens of graph models. Graph models works include, among the others, Graph-Based Models [Kipf and Welling, 2017], Layer-wise Relevance Propagation [Montavon *et al.*, 2019], and Complex Network Theory [Boccaletti *et al.*, 2006]. In particular, Complex Network Theory (CNT) is a branch of mathematics that represents complex systems, from city connectivity to networks of computers [Porta *et al.*, 2006b], by modelling and then simulating their dynamics through graphs where nodes represent entities and vertices relationships [Crucitti *et al.*, 2004; Porta *et al.*, 2006a; Chavez *et al.*, 2010]. CNT offers an intuitive method for conceptualising DNNs, where neurons are analogous to graph nodes and connections to weighted edges.

This paper characterises DNNs as graphs via CNT metrics: we develop and unify a set of metrics that describe a network's weights, neurons, and the hidden layers' behaviour: we uncover consistent trends across various architectures, initialisations, and objective tasks across range of network architectures, including Fully Connected (FC), AutoEncoders (AE), Convolutional (CNNs), and Recurrent Neural Networks (RNN). The visual representations of the CNT metrics provide insights into a DNN's decision process. In synthesis, our work formally connects and grounds the existing work in the field [Testolin *et al.*, 2019; Scabini *et al.*, 2022] by unifying and extending existing CNT metrics such as *Link Weigths Dynamics*, *Nodes Strength* and *Layers Fluctuation* [La Malfa *et al.*, 2021] moving beyond a topological analysis to account for the effect of the input data. Section 2 surveys the literature of CNT applied to DNNs. In Section 3, we introduce our methodology, i.e., a unified framework to study DNNs via CNT metrics and how to compute the various metrics for different DNN architectures. Section 4 reports results for different architectures, activation functions, and shallow and deep networks. We conclude the article with some ideas for further development and future works in this raising area of research.

2 Related Works

Testolin et al. [2019] first used CNT to interpret deep learning models on classification tasks. Their work leverages CNT to retrieve information from Deep Belief Networks, a generative model whose unsupervised learning phase differs from feed-forward neural networks. They conducted experiments on MNIST and discovered a tension between elicited and suppressed neurons when classifying digits and that suppression is correlated with the efficiency of a network on the classification task. Zambra et al. [2020] moved beyond neuron activations and studied DNNs connected components, or 'motifs' as connectivity patterns between neurons or layers. 'Motifs' shed light on how a DNN learns and generalises. They discovered that initialising a neural network weights is key to the emergence of such 'motifs' and connects to accurate learning. In their study, Petri et al. [2021] show the inherent trade-offs between the capacity of a DNN to learn, generalise and execute multiple tasks simultaneously. This finding underscores a critical challenge in developing versatile and efficient AI systems unveiled through the analysis of neural networks using topological representations, an approach akin to CNT that examines a network structure and connectivity. In continuity with Petri et al., Saxe et al. [2022] introduce the Gated Deep Linear Network model. Through this model, they reveal that the learning process in structured networks is a 'neural race' in which different components compete to learn and represent information. This process is biased towards forming structured representations of knowledge, which determine a DNN's ability to generalise, transfer knowledge and handle multiple tasks simultaneously. La Malfa et al. [2021] study the DNNs training dynamics through the lens of CNT. Their approach is local and global in that it studies single networks and populations of evenly initialised DNNs. They formalise metrics for weights, neurons and hidden layers, yet their measures do not account for the value of the input. Scabini et al. [2022] unveil the topological properties of a broad range of FCs and the benefits of random weight initialisation. They introduce Bag-Of-Neurons, a technique designed to identify topological signatures to group similar neurons (i.e., those dedicated to solving a specific sub-problem of a task such as edge detection).

3 Methodology

First, we provide background notation to describe a DNN via CNT metrics [Testolin *et al.*, 2019; La Malfa *et al.*, 2021]: the reference architecture is an FC network that solves a classification task. We then introduce CNT metrics that account for the input data via sampling from the training distribution. Finally, we show how to compute these metrics for a broad range of neural network architectures, including CNNs and RNNs. We also briefly discuss how attention networks [Vaswani *et al.*, 2017] can be studied as Complex Networks, though we do not cover Transformers as we believe, given their complex architecture, they constitute a separate work.

3.1 Background

DNNs as graphs have three main components: the *input layer*, which is the 'receptor' of the input data (e.g., pixels of an image, audio samples, textual data, etc.), the *hidden layers*, which are stacked layers of neurons that transform the input via affine transformation followed by non-linear function activations, and the *output*, e.g., the network's classification. Formally, we consider an FC network that solves a supervised classification task, i.e., the network learns an input-output mapping $f : \mathbb{R}^d \rightarrow \mathbb{R}^m$ that minimises a generic loss function $\mathcal{L}(f(x), y)$, between each input-output pair (x, y). An input x is a d-dimensional vector $x \in \mathbb{R}^d$ drawn from a distribution, while each corresponding output is either from a discrete set in case of classification, i.e., $c \in C \;.\; |C| = m$, or it is continuous in case of regression, i.e., $y \in \mathbb{R}^m$. An FC architecture consists of $L > 0$ dense layers stacked together, each of a variable number of neurons: within each hidden layer ℓ, a neuron $n_i^{[\ell]}$ is connected through a weighted link to all the neurons of the successive layer $\ell + 1$. The output $z^{[\ell]}$ of a layer ℓ is the product of an affine transformation between a matrix of weights $\Omega^{[\ell]}$, plus eventually a bias term $\beta^{[\ell]}$, namely $z^{[\ell]} = z^{[\ell-1]}\Omega^{[\ell]} + \beta^{[\ell]}$, followed by a non-linear activation function $f^{[\ell]}(z^{[\ell]})$. For an FC network, $\Omega^{[\ell]}$ is a matrix of size $\mathcal{N}^{[\ell]} \times \mathcal{N}^{[\ell+1]}$ and $\beta^{[\ell]}$ is a vector of size $\mathcal{N}^{[\ell+1]}$. We denote the input and output vectors as $x = z^{[0]}$ and $y = z^{[L]}$, while $\Omega^{[\ell]}$ and $\beta^{[\ell]}$ refer to the parameters of a neural network layer ℓ. The output of the neural network is defined as $y = f^{[L]}(z^{[L]}) = z^{[L-1]}\Omega^{[L]} + \beta^{[L]}$. For a classification task, the output is a vector of real numbers $y \in \mathbb{R}^m$, from which the $argmax$ operator extracts the predicted class. We refer to the input-output relation of a neural network at layer ℓ as $z^{[\ell]} = \mathbf{f}(x,\ \Omega^{[:\ell]},\ \beta^{[:\ell]})$.

3.2 Complex Networks Metrics for DNNs

A DNN can be represented as a set of nodes (neurons) connected by weighted edges. This intuition is sufficient to describe a DNNs via CNT. Formally, a DNN is a directed bipartite graph $G = \langle N, E \rangle$, where each node $n_i^{[\ell]} \in N$ corresponds to a neuron in the ℓ-th hidden layer. The intensity of a connection is a real number $\omega_{i,j}^{[\ell]}$ assigned to an edge ($e_{n_i^{[\ell]}, n_j^{[\ell+1]}} \in E$) that connects two neurons.

Link Weights. The Link Weights provide insight into how weights and biases adapt during training. Standard measurements of such metrics are the weights mean and variance at each layer during training. For DNN at layer ℓ they are defined as:

$$\mu^{[\ell]} = \frac{1}{N^{[\ell]} N^{[\ell+1]}} \sum_{i=1}^{N^{[\ell]}} \sum_{j=1}^{N^{[\ell+1]}} \omega_{i,j}^{[\ell]} + \beta_i^{[\ell]} \tag{1}$$

$$\delta^{[\ell]} = \frac{1}{N^{[\ell]} N^{[\ell+1]}} \sum_{i=1}^{N^{[\ell]}} \sum_{j=1}^{N^{[\ell+1]}} ((\omega_{i,j}^{[\ell]} + \beta_i^{[\ell]}) - \mu^{[\ell]})^2 \tag{2}$$

Monitoring weights mean and variance throughout training provides insights into the learning process's effectiveness and stability. If the norm of the weights does not increase, it could indicate over-regularisation in the model. Conversely, excessively large weight values might lead to overfitting.

Nodes Strength. Formally, the strength $s_k^{[\ell]}$ of a neuron $n_k^{[\ell]}$ is the sum of the weights of the edges incident in $n_k^{[\ell]}$. Since neural network graphs are directed, two components contribute to the Node Strength: the sum of the weights of outgoing edges $s_{out,k}^{[\ell]}$, and the sum of the weights of in-going links $s_{in,k}^{[\ell]}$.

$$s_k^{[\ell]} = s_{in,k}^{[\ell]} + s_{out,k}^{[\ell]} = \sum_{i=1}^{N^{[\ell]}} (\omega_{i,k}^{[\ell]} + \beta_k^{[\ell]}) + \sum_{j=1}^{N^{[\ell+1]}} \omega_{k,j}^{[\ell+1]} \quad (3)$$

Node strength reflects how strong a connection is for a specific feature. In literature, the Node Strength account for both in- and out-coming edges [La Malfa *et al.*, 2021]; we will consider them separately.

Layers Fluctuation. *Layers Fluctuation* extends the idea of Nodes Fluctuation [Porta *et al.*, 2006b] to measure the variability of metrics at the level of a network's hidden layers. CNT classically defines Nodes Disparity as $Y^{[\ell]} = \sum_{i=1}^{N^{[\ell]}} [\omega_i^{[\ell]} / s_i^{[\ell]}]^2$. However, weights generally have positive and negative values that could cancel each other out. To address this issue, we propose a metric designed for DNNs that captures the strength fluctuations within each layer, reflecting the interactions among nodes at the same depth. The Layers Fluctuation is defined as:

$$Y^{[\ell]} = \sqrt{\frac{\sum_{i=1}^{N^{[\ell]}} (s_i^{[\ell]} - \hat{s}^{[\ell]})^2}{I}} \quad (4)$$

where $\hat{s}^{[\ell]}$ is computed as the average value of Nodes Strength at layer ℓ, namely $\hat{s}^{[\ell]} = \frac{1}{N^{[\ell]}} \sum_{i=1}^{m} s_i^{[\ell]}$, being $N^{[\ell]}$ the number of nodes/neurons at layer ℓ. This metric identifies asymmetries and disparities in the network at the layer level rather than focusing on individual nodes. Unlike the standard Nodes Fluctuation, Layers Fluctuation characterises the dynamics of an entire layer and accounts for bottlenecks within the network architecture.

3.3 Data-dependent CNT Metrics

We now introduce and formalise CNT metrics that account for the value of the input data, namely the *Neurons Strength* and *Neurons Activation*. Specifically, the training data distribution informs the computation of *Neurons Strength* and *Neurons Activation* so that for two different datasets, the results would vary accordingly (while it is not the case for the previous metrics). We then discuss how to compute all the above CNT metrics, including *Nodes Strength* and *Layer Fluctuation* to CNNs and RNNs.

Neurons Strength. The mathematical formulation of Neurons Strength for a neuron $n_k^{[\ell]}$ in layer ℓ is given by:

$$\zeta_k^{[\ell]} = \sum_{i=1}^{N^{[\ell]}} z_i^{[\ell-1]} \omega_{i,k}^{[\ell]} + \beta_k^{[\ell]},$$
$$z^{[\ell-1]} = \mathbf{f}(x, \Omega^{[:\ell]}, \beta^{[:\ell]}) \; . \; x \sim \mathcal{X} \quad (5)$$

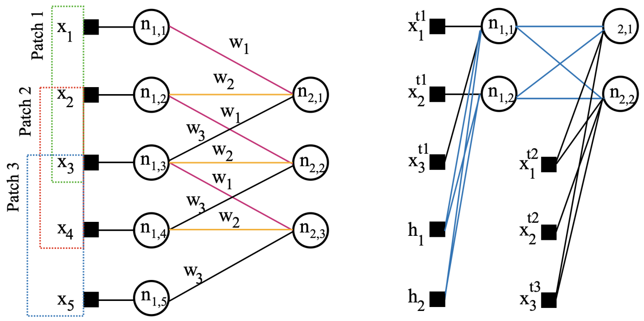

Figure 1: For CNNs, CNT metrics are computed by isolating each input patch and the kernel responsible for a dot-product in a layer (left), while for RNNs, metrics can be computed by unfolding each input feature through time (right).

Here, $\zeta_k^{[\ell]}$ represents the strength of neuron k in layer ℓ, considering the effects of both the activation functions of previous layers and the input values drawn from a distribution $\mathcal{X}$. This approach provides a more comprehensive understanding of the neuron's role and influence within the network, factoring in the data being processed.

Neurons Activation. In Deep Neural Networks (DNNs), each neuron's activation level is determined by both the input values and the specific activation functions used in the network. The activation of a neuron in layer ℓ can be mathematically expressed as:

$$a_k^{[\ell]} = f^{[\ell]}(\sum_{i=1}^{N^{[\ell]}} z_i^{[\ell-1]} \omega_{i,k}^{[\ell]} + \beta_k^{[\ell]}),$$
$$z^{[\ell-1]} = \mathbf{f}(x, \Omega^{[:\ell]}, \beta^{[:\ell]}) \; . \; x \sim \mathcal{X}. \quad (6)$$

This equation highlights how the activation value $a_k^{[\ell]}$ of a neuron depends on the weighted sum of activations from the previous layer, adjusted by the neuron's weights and bias, and then transformed by the activation function $f^{[\ell]}$. A neuron exhibiting an unusually high Node Strength transmits a stronger *signal* than others. In such a scenario, an input does not elicit all the neurons uniformly, i.e., a neuron conveys more significant information for the classification task. Conversely, a neuron that transmits a weak *signal* might be a candidate for pruning, which can help reduce the overall complexity of the network without significantly impacting the layer's output.

3.4 CNT Metrics: Beyond Fully Connected Layers

The role of architectural inductive biases in enhancing the performance of deep learning models is well-established and supported by decades of extensive research. These biases, integrated into a model architecture, have significantly influenced the field of artificial intelligence. For instance, CNNs were designed with biases towards local connectivity, inspired by the human visual system [Lecun and Bengio, 1995]. Similarly, the development of recurrent networks, particularly the introduction of gates and memory cells in Long Short-Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997], exemplifies how these biases enable the retention and accessibility of information over extended time steps. In this section, we adapt CNT metrics to CNN and RNN architectures, thus moving beyond the model of reference in most works on CNT applied to DNNs. We also discuss self-attention, the building block of Transformers [Vaswani *et al.*, 2017], and how its architecture can be reconciled with one amenable to be studied with CNT. However, we leave the formalisation and the experiments for Transformers as future work.

Convolutional Neural Networks. Convolution is the building block operator in CNNs. An input matrix of size $w \times h$ (e..g, an image represented as a grid of pixels) is split into (possibly overlapping) patches of size $k \times k$. A point-wise multiplication between each patch and the kernel, which has the same size as each patch, shrinks the input and produces an output that is then activated and fed to the next DNN's layer. In mathematical terms, for an input matrix z_ℓ of dimensions $w \times h$ and a kernel Ω of size $m \times m$ (with $m \leq w$ and $m \leq h$), the convolution operation can be represented as $z' = z * \Omega$. We now report the formula to compute the Neurons Strength for a CNN layer to highlight the differences with the Fully Connected case. For the Neurons Activation, the operations are analogous.

$$\zeta_k^{[\ell]} = z_k^{[\ell-1]} * \Omega^{[\ell]} + \beta_k^{[\ell]}, \qquad z^{[\ell-1]} = \mathbf{f}(x, \Omega^{[:\ell]}, \beta^{[:\ell]}) \,.\, x \sim \mathcal{X}. \quad (7)$$

In the previous equation, $z_k^{[\ell-1]} * \Omega^{[\ell]}$ represents the convolution between the input patch $z_k^{[\ell-1]}$ and the entire kernel of weights at layer ℓ. One straightforward method to apply CNT metrics to this convolution operation is to transform the convolution into a dot product operation using a Toeplitz matrix. However, while this method is conceptually simple, it significantly increases both the time and space complexity from linear to quadratic, and it is thus expensive for large networks. Our approach, therefore, is to tackle this challenge efficiently. We isolate each input portion that is element-wise multiplied by the kernel. By linking each input neuron to its corresponding output neuron (as illustrated in Figure 1, left), we can efficiently compute the CNT metrics for any layer. This method allows to apply complex network analysis to convolution operations without the prohibitive computational cost of the Toeplitz matrix approach.

Recurrent Neural Networks. RNNs are designed to process sequential input, where the output of the previous step influences the output at each step in the sequence. Recursion makes RNNs ideal for tasks involving temporal data, like speech recognition or language modelling. The output of a single-layer recurrent RNN for the $t > 0$ temporal dimension of x is the following:

$$h^{(t+1)} = \mathbf{f}(z^{(t)}\Omega + h^{(t)}U). \quad (8)$$

In the previous equation, where the bias term is omitted for clarity, U represents a matrix of trainable parameters used to process the hidden unit $h^{(t)}$. Usually the value of $h^{(1)}$ is initialised to zero. To adapt CNT metrics for RNNs, we 'unfold' each input feature along its temporal dimension and treat each recurrent step as a layer-wise multiplication in an FC topology. This transformation is computationally less efficient than relying on symbolic loops, yet it is necessary to compute the CNT metrics for each input feature. The closed form formula to compute the Neurons Activation for an RNN Cell with one hidden layer, at time $t > 0$, and with parameters Ω and U (as sketched in Figure 1, right) is the following:

$$\zeta^{(t)} = x^{(t-1)}\Omega + \mathbf{f}(\dots \mathbf{f}(x^{(1)}\Omega + h^{(1)}U)\dots)U \,.\, x \sim \mathcal{X}. \quad (9)$$

In the previous equation, the value of the Neuron Strength is computed by first sampling an input x from the input data distribution $\mathcal{X}$, then by recursively unrolling the RNN and processing each temporal feature $x^{(j)}$ to compute the value of the hidden unit $h^{(j)}$ at times $j < t$. In the experiments, we show how the Neuron Strength can enhance a model's interpretability of which input features elicit the most neurons in an RNN recursive layer.

Self-attention. We conclude the section with a concise discussion of how self-attention can be expressed in a form amenable to study with CNT metrics. Self-attention (the building block of Transformers) is mathematically expressed as $z = \text{softmax}(\langle xW^K, xW^Q \rangle)xW^V$, where $\langle xW^K, xW^Q \rangle$ is the dot-product between a *key* and a *query*.[1] Being the key-query a dot-product between two tensors, it can be expressed as an operation between nodes in a Complex Network and thus studied via CNT. The same argument is valid for multiplying the attention scores with xW^V. We leave the formalisation of Transformers, the building block of modern Large Language Models, and the experiments to interpret their internal to future works.

4 Experiments

In this section, we conduct experiments to assess to which extent CNT identifies patterns in DNNs: we define three complementary levels of analysis. The first level (I) aims to distinguish dominating CNT patterns for architecturally similar networks: we train on MNIST and CIFAR10 three-layer depth FCs, CNNs, RNNs and AEs equipped with the same activation functions and a comparable number of parameters. While we test FCs, CNNs, and RNNs on image classification, AEs are trained to compress and reconstruct the input. As we keep the architectures as simple as possible, MNIST results in a relatively simple task where all the models perform well, while on CIFAR10, CNNs have better performances aided by their inductive bias towards image classification. The second level (II) studies the incidence of different activation functions (i.e., linear, ReLU, sigmoid) on CNT metrics for FCs, CNNs, RNNs and AEs. Similarly to the previous setting, the networks are architecturally similar in terms of hidden layers and number of trainable parameters, yet differ in their activation functions. The third level of analysis (III) explores the impact of depth (i.e., the number of hidden layers) on CNT metrics. Deeper neural networks learn more complex features

[1] The normalisation factor inside the softmax can be, without loss of generality ignored.

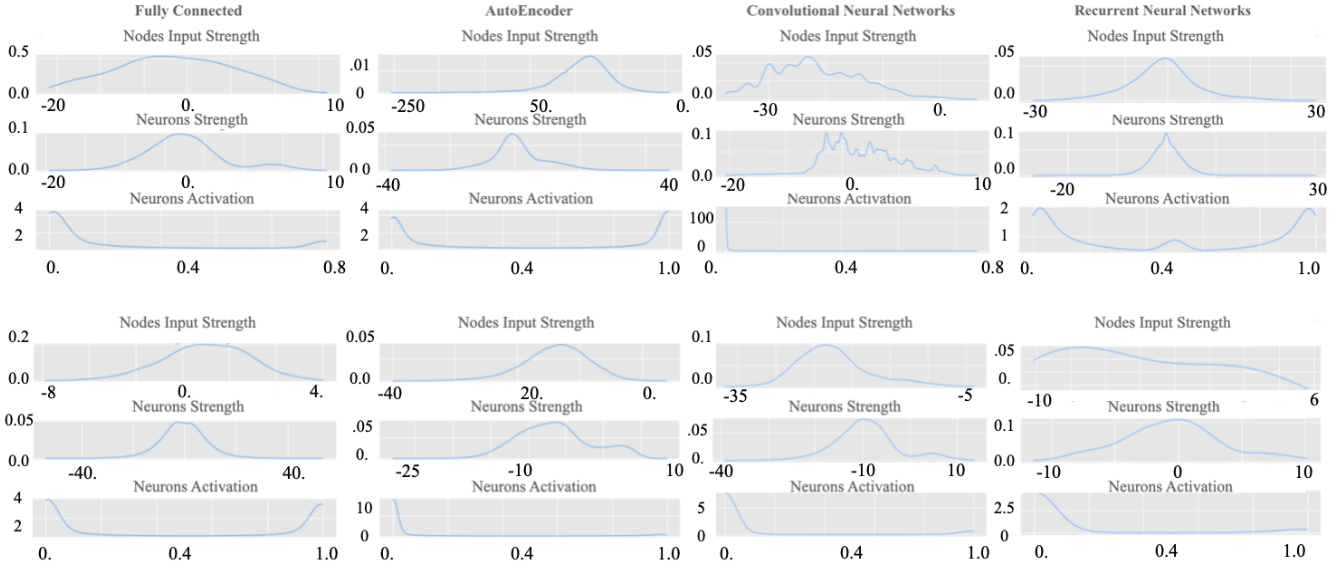

Figure 2: Analysis of CNT metrics across the second and third layers (hidden and output layers) for three-layer depth FCs, CNNs, RNNs and AEs on the MNIST dataset. Each column corresponds to an architecture, and the figures illustrate the distribution functions computed on a pool of 30 neural networks trained on the task.

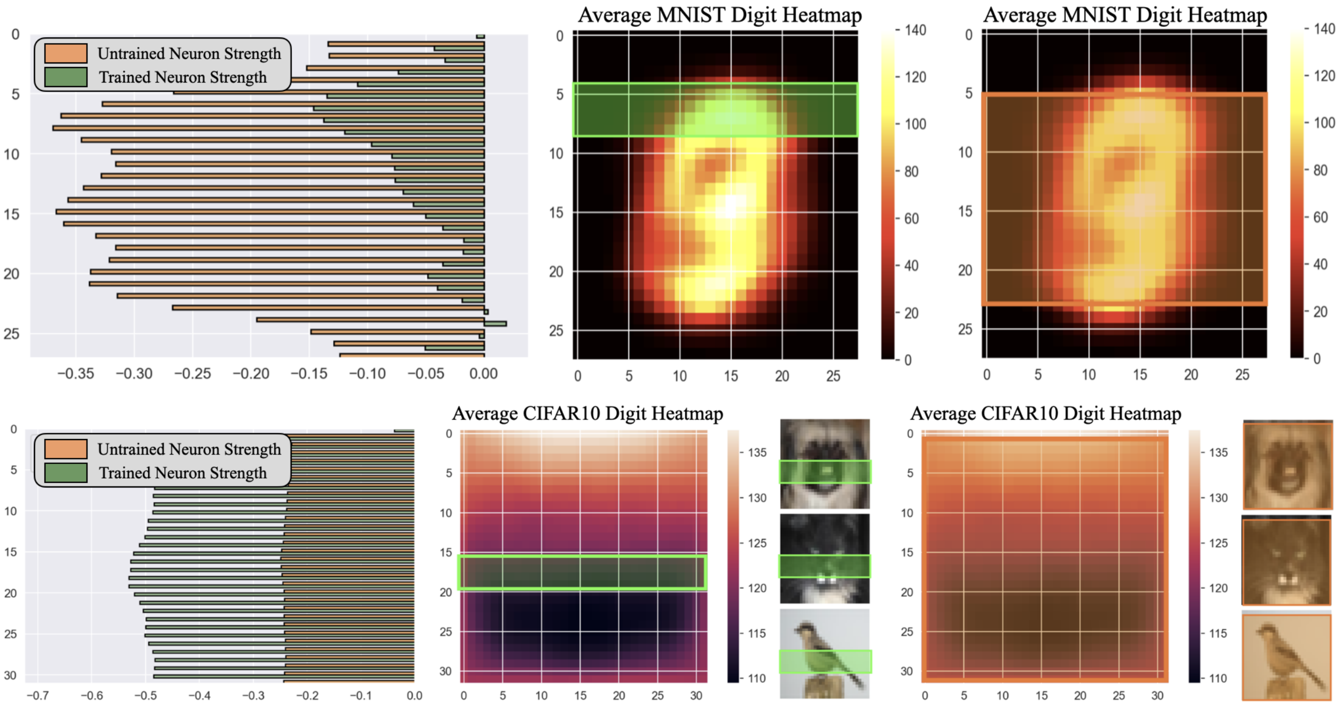

Figure 3: Neurons Strength metric for 30 RNNs on MNIST (top) and CIFAR10 (bottom) classification tasks, distinguishing between networks that have been trained and those that remain untrained. The left side of the figure quantifies the Neurons Strength, while the right side visualises a global heatmap of neurons that are most elicited by MNIST/CIFAR10 inputs.

from data and positively correlate with a DNN's performance on the task. Also, deeper layers can learn increasingly abstract representations of the data. In image processing, initial layers might detect edges and textures, while deeper layers might identify more complex patterns or objects. We investigate and report the impact of depth by varying the number of hidden layers of FCs and AEs from three to seven (while further analyses are reported in the code repository).

We conduct all the experiments on two standard datasets in pattern recognition and computer vision, namely MNIST and

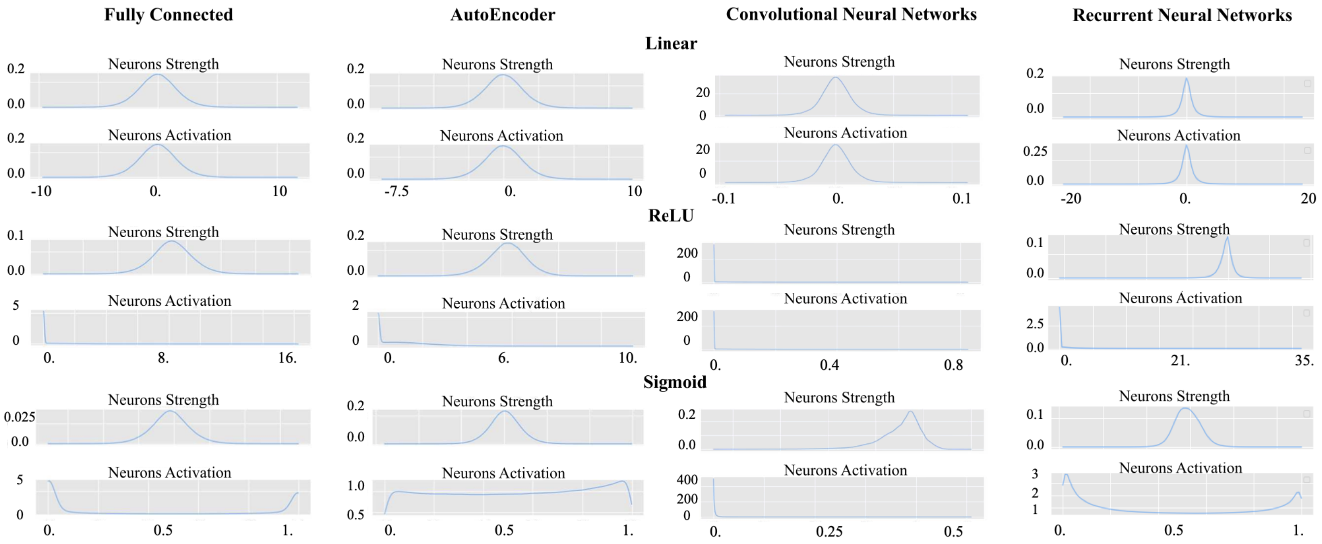

Figure 4: Analysis of CNT metrics for three-layer depth FCs, CNNs, RNNs and AEs on the CIFAR10 dataset and different activation functions (linear, ReLU and sigmoid). Each column corresponds to an architecture, and the figures illustrate the distribution functions computed on a pool of 30 neural networks trained on the task.

CIFAR10 [Lecun and Bengio, 1995; Krizhevsky *et al.*, 2010]. While relatively simple, these two benchmarks set a baseline for future analyses on more complex tasks; we stress that our framework can be applied to any dataset a DNN can tackle. Generally, we compute all the CNT metrics on a 30 trained neural network pool. We initialise the weights of each DNN via sampling from a Gaussian distribution of known variance between 0.05 (MNIST) and 0.5 (CIFAR10).[2]

I. CNT Metrics of different architectures. MNIST images consist of white-scaled digits on a dark background. As reported in the literature, FC networks tend to present 'dead units', i.e. an abundance of negative or close to zero weights, a hint of an over-parametrised network [Testolin *et al.*, 2019]. Our results confirm this trend for FCs and extend it to all the other architectures (CNNs, RNNs and AEs), as we show in Figure 2; yet, for AEs, the phenomenon is more remarked, with a probability distribution of Neurons Strength and Activation peaked respectively around negative values and at the extremes of the distribution. We further notice that AEs compress information by design, i.e., a 'bottleneck', as shown in the third layer of Figure 2. On the other hand, CNNs' Nodes Strength and Neurons Strength are multimodal, with spikes corresponding to the patterns a network learns to perform local edge detection. We hypothesise that CNNs specialise neurons within the same layer to activate them for different input values. For example, the spikes may represent the information relative to the number of edges in a digit, as that is one of the sub-tasks that requires more information to be encoded. Globally, the influence of positive and negative weights in FC, AEs and RNNs seems balanced, while CNNs exhibit a profusion of negative weights. This suggests that, during the training phase, CNNs may prioritise different input features than FCs, RNNs and AEs. RNNs' Neuron Strength distribution exhibits a pronounced Kurtosis, a feature that isn't observed when analysing Node Input Strength alone and is thus imputable to the data distribution. Additionally, the Neuron Activation displays a unique pattern: unlike other architectures, the density is distributed not just at the extremes of the sigmoid function, but also significantly in the middle (between 0.4 and 0.6), a sign that recurrent architectures are less likely to settle at the extremes, resulting in more varied activation patterns. Regarding interpretability, we further investigate the dynamics of the Neurons Strength in an RNNs' autoregressive loop. This approach involves an RNN sequentially processing a segment of the input, updating its hidden states, and ultimately classifying the image. In Figures 3, we compare the Neurons Strength of trained and untrained RNNs on the MNIST and CIFAR10 datasets. In both cases, the Neurons Strength of trained RNNs localises a specific image region that elicits the neurons the most, while that doesn't occur for untrained networks. This suggests that the training phase calibrates a model to activate mostly on specific patterns located in the central part of the input. We compare our results with that of Layer-Wise Relevance Propagation (LRP) [Bach *et al.*, 2015] on a sample of inputs of the same class as those in Figure 3 (top), a standard interpretability framework that identifies the most activated neurons at inference time and maps them to specific pixels in the input region. We report the interpretability heatmaps for different LRP implementations in Figure 6 on the MNIST dataset. Unlike CNT metrics, LRP is local (i.e., it allows determining the salient features of a single input point), works backwards (from the output to the input), and covers only neurons. Our techniques are global and local (they allow us to analyse single and multiple data points), work forward (neural networks are not invertible, which causes a loss of information in LRP), and extend

[2] Higher variance in the parameter initialisation can sometimes help the network to escape poor local minima early in training, which are more prevalent in datasets like CIFAR10.

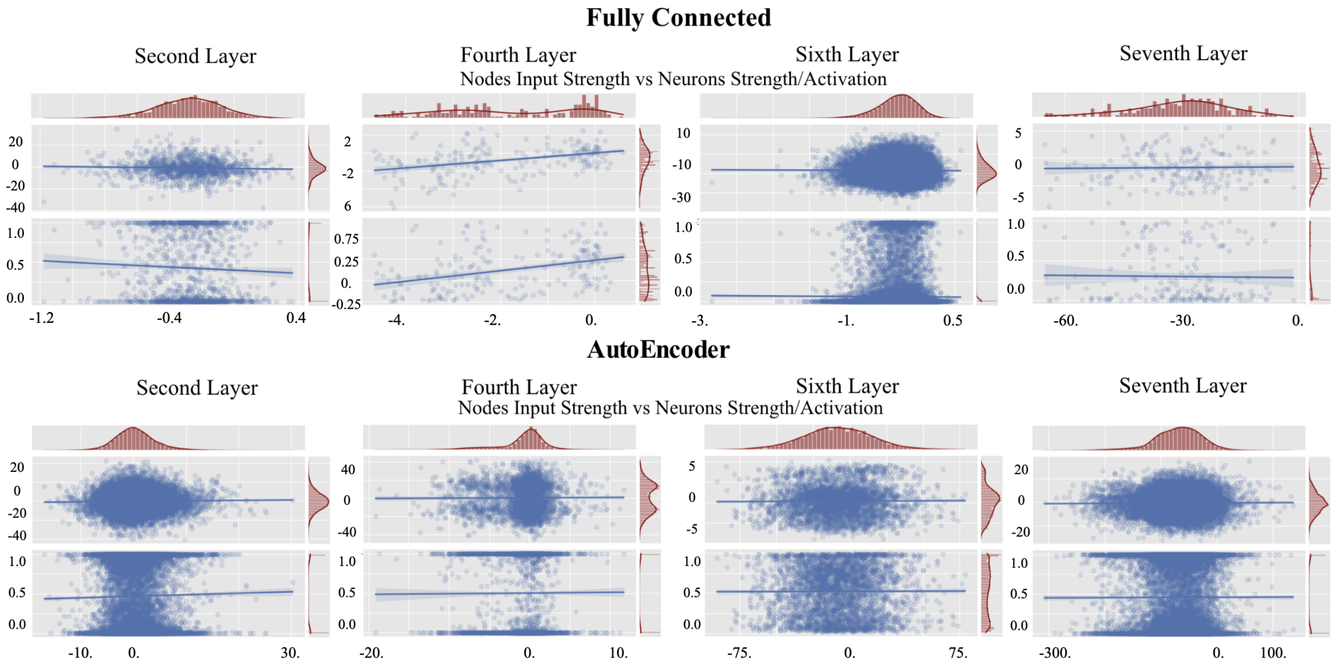

Figure 5: Neurons Strength and Activation, and scatter-plot of the correlation between Nodes Strength and Neurons Strength and Activation, for seven-layer depth FCs and AEs. The figures illustrate the distribution functions computed on a pool of 30 neural networks trained on the task.

to weights, neurons, and layers. Interestingly, LRP and CNT metrics identify different activation patterns in the input. For the MNIST dataset, LRP usually identifies the receptive field of the input numbers. At the same time, CNT discriminates based on the upper region of each image, a result that requires further investigation in future works.

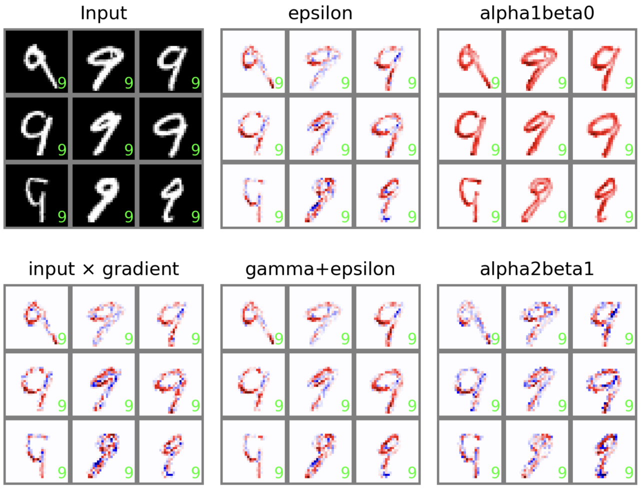

Figure 6: Activation patterns of variations of LRP implementations on inputs from the MNIST that belong to class *nine* (as per Figure 3, top). Credits for the implementation to TorchLRP.

II. Incidence of the activation function. Non-linear activation functions allow models to learn complex patterns in data. We compare the effects of linear, ReLU, and sigmoid activation functions for FCs, CNNs, RNNs and AEs on CIFAR10. Linear activations, lacking the ability to model non-convex optimisation landscapes, typically lead to Neurons Strength and Neurons Activation distributions centred around zero. In contrast, ReLU and sigmoid activations result in asymmetric distributions, favouring negative values, often correlating with enhanced accuracy. Notably, ReLU's inherent asymmetry results in distributions skewed more significantly towards extreme negative values than those observed with sigmoid, as we report in Figure 4 (first column). AEs with linear and ReLU activations show similar behaviours to FCs. However, models equipped with sigmoid exhibit Neuron Activation distributed evenly around its support. This phenomenon suggests the AEs correctly reconstruct the input data with dynamics that diverge from FCs, CNNs and RNNs, which solve a different task (classification). For linear models, CNNs' Neurons Activation centres around zero, with low variance compared to other architectures. Conversely, non-linear activations skew the distribution of the metrics toward zero. Both ReLU- and sigmoid-activated networks leverage negatively activated neurons to discriminate between different classes. This phenomenon is remarkably different from what happens with FCs and RNNs where the Neurons Activation accumulates around the extreme values of the activation (zero and one), suggesting that the decision rules encoded internally by CNNs may diverge from that of other architectures.

III. Incidence of neural network depth. We conclude with an analysis of the effect of depth on the dynamics of

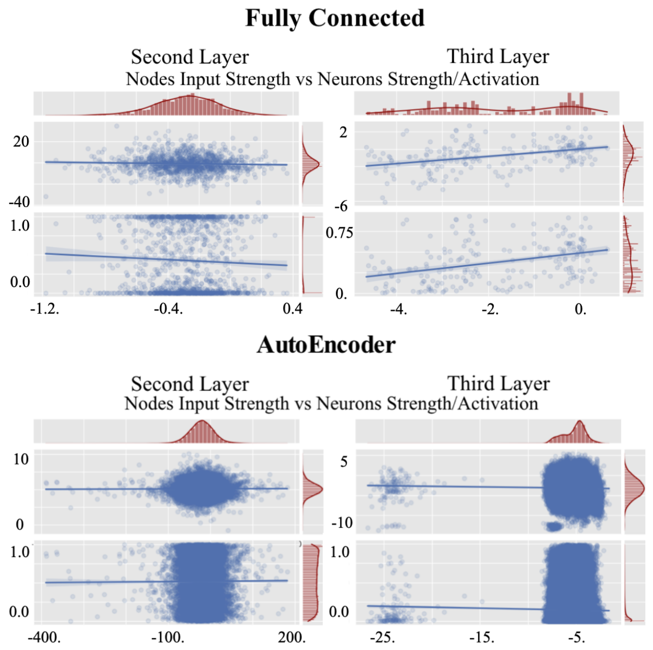

Figure 7: Neurons Strength and Activation, and scatter-plot of the correlation between Nodes Strength and Neurons Strength and Activation, for three-layer depth FCs and AEs. The figures illustrate the distribution functions computed on a pool of 30 neural networks trained on the task.

CNT metrics on CIFAR10. We compare FCs and AEs with three and seven layers, respectively. While FCs perform classification, AEs are trained to reconstruct the input with minimal data loss.[3] In Figure 7, we report the Neurons Strength and Activation, which we further correlate with the Nodes Strength. Shallow FC architectures operate with similar dynamics as deeper networks, with the second and third layers of Figure 7 (top) that overlap with respectively the fourth and seventh of Figure 5 (top): in conclusion, deeper networks exhibit more complex patterns in their hidden layers, that shallow lack. For this task, deeper networks leverage the same 'building block' of shallow networks to build increasingly complex input representations and solve the task more accurately. Conversely, the dynamics of shallow and deep AEs diverge sensibly: seven-layer AEs seem not to leverage existing 'building blocks' of shallow architectures, suggesting that data compression is a dynamic process heavily influenced by the number of layers in a network. We also notice that the Nodes Strength does not correlate with the Neurons Strength or Activation, supporting the development of CNT metrics that incorporate the effect of data in their dynamics.

5 Conclusions and Future Works

This paper introduces a unified framework for representing neural networks via CNT, incorporating new metrics influenced by input data and enhancing traditional topological CNT analysis. Our extensive experiments on FCs, CNNs, RNNs, and AEs reveal distinct dynamics and training patterns across architectures, activation functions, and depths. our key findings include over-parametrisation in most architectures, especially with the MNIST task, and CNNs demonstrating localised learning through multimodal Nodes and Neurons Strength. Non-linear activations in models lead to asymmetric distributions, causing complex pattern learning. AEs' Neurons Activation distribution in deeper models suggests their learning dynamics do not leverage the same dynamics as shallow architectures. Unlike FCs and RNNs, CNNs with non-linear activations learn discriminative patterns for classification, mostly in negative regions of the activation function. In future works, we will extend this framework to advanced architectures, both theoretically and empirically. In particular, we believe that CNT can contribute to interpreting self-attention [Vaswani *et al.*, 2017], an architecture that powers language and computer vision models. This work is also meant to encourage other researchers with expertise in both machine learning and physics to contribute to the formalisation of learning systems as graphs to unveil their training dynamics as the current state-of-the-art approaches do not study the evolution of untrained networks during the learning phase.

Acknowledgments

G.L.M.'s work was supported by UK Research and Innovation [grant number EP/S023356/1], in the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence (www.safeandtrustedai.org).

G.N. thanks the University of Cambridge for the extended research periods and the opportunity to be a Visiting Professor during the Sabbatical year. Moreover, G.N. thanks Wolfson College, the Department of Biochemistry, and the Cambridge Systems Biology Centre for the hospitality and warm welcome that allowed for peaceful studies, novel, interesting collaborations, and fruitful research activities.

V.L. acknowledges support from the European Union, NextGenerationEU, GRINS project (GRINS PE00000018 - CUP E63C22002120006).

References

[Bach *et al.*, 2015] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[Bishop, 1995] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, USA, 1995.

[Boccaletti *et al.*, 2006] S. Boccaletti, V. Latora, Moreno Y, M. Chavez, and D. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4-5):175–308, February 2006.

[Chavez *et al.*, 2010] Maisa Chavez, Miguel Valencia, Valy Navarro, Vito Latora, and J. Martinerie. Functional modularity of background activities in normal and epileptic brain networks. *Physical Review Letters*, 104(11), March 2010.

[3] Results for the other architectures on MNIST and CIFAR10, for five and nine layers, are reported in the code repository.

[Crucitti *et al.*, 2004] Paolo Crucitti, Vito Latora, and Massimo Marchiori. A topological analysis of the italian electric power grid. *Physica A: Statistical Mechanics and its Applications*, 338(1-2):92–97, July 2004.

[Erhan *et al.*, 2009] Dumitru Erhan, Y. Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Technical Report, Univeristé de Montréal*, 01 2009.

[Ghorbani *et al.*, 2019] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3681–3688, 2019.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Karpathy *et al.*, 2015] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks, 2015.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[Krizhevsky *et al.*, 2010] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5(4):1, 2010.

[La Malfa *et al.*, 2021] Emanuele La Malfa, Gabriele La Malfa, Giuseppe Nicosia, and Vito Latora. Characterizing learning dynamics of deep neural networks via complex networks. In *2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 344–351. IEEE, 2021.

[Lecun and Bengio, 1995] Yann Lecun and Yoshua Bengio. *Convolutional networks for images, speech, and time-series*. MIT Press, 1995.

[Montavon *et al.*, 2018] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital signal processing*, 73:1–15, 2018.

[Montavon *et al.*, 2019] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. *Layer-Wise Relevance Propagation: An Overview*, pages 193–209. Springer International Publishing, Cham, 2019.

[Petri *et al.*, 2021] Giovanni Petri, Sebastian Musslick, Biswadip Dey, Kayhan Özcimder, David Turner, Ahmed Nesreen K., Willke Theodore L., and Cohen Jonathan D. Topological limits to the parallel processing capability of network architectures. *Nature Physics*, 17(5):646–651, May 2021.

[Porta *et al.*, 2006a] Sergio Porta, Paolo Crucitti, and Vito Latora. The network analysis of urban streets: A dual approach. *Physica A: Statistical Mechanics and its Applications*, 369(2):853–866, September 2006.

[Porta *et al.*, 2006b] Sergio Porta, Paolo Crucitti, and Vito Latora. The network analysis of urban streets: a primal approach. *Environment and Planning B: planning and design*, 33(5):705–725, 2006.

[Saxe *et al.*, 2022] Andrew Saxe, Shagun Sodhani, and Sam Jay Lewallen. The neural race reduction: Dynamics of abstraction in gated networks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 19287–19309. PMLR, 17–23 Jul 2022.

[Scabini *et al.*, 2022] Leonardo Scabini, Bernard De Baets, and O. M. Bruno. Improving deep neural network random initialization through neuronal rewiring, 2022.

[Schmidhuber, 2015] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[Singh *et al.*, 2018] Chandan Singh, W James Murdoch, and Bin Yu. Hierarchical interpretations for neural network predictions. *arXiv preprint arXiv:1806.05337*, 2018.

[Testolin *et al.*, 2019] A. Testolin, M. Piccolini, and S. Suweis. Deep learning systems as complex networks. *Journal of Complex Networks*, June 2019.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[Zambra *et al.*, 2020] Matteo Zambra, Amos Maritan, and Alberto Testolin. Emergence of network motifs in deep neural networks. *Entropy*, 22(2):204, February 2020.

[Zeiler and Fergus, 2014] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833. Springer International Publishing, 2014.