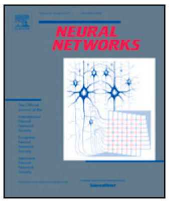
Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



Full Length Article

Beyond multilayer perceptrons: Investigating complex topologies in neural networks

Tommaso Boccato [a,*], Matteo Ferrante [a], Andrea Duggento [a,1], Nicola Toschi [a,b,1]

[a] Department of Biomedicine and Prevention, University of Rome Tor Vergata, Rome, Italy
[b] A.A. Martinos Center for Biomedical Imaging and Harvard Medical School, Boston, USA

ARTICLE INFO

Dataset link: https://github.com/BoCtrl-C

Keywords:
Neural networks
Complex networks
Bio-inspired computing
Manifold learning
Robustness

ABSTRACT

This study delves into the crucial aspect of network topology in artificial neural networks (NNs) and its impact on model performance. Addressing the need to comprehend how network structures influence learning capabilities, the research contrasts traditional multilayer perceptrons (MLPs) with models built on various complex topologies using novel network generation techniques. Drawing insights from synthetic datasets, the study reveals the remarkable accuracy of complex NNs, particularly in high-difficulty scenarios, outperforming MLPs. Our exploration extends to real-world datasets, highlighting the task-specific nature of optimal network topologies and unveiling trade-offs, including increased computational demands and reduced robustness to graph damage in complex NNs compared to MLPs. This research underscores the pivotal role of complex topologies in addressing challenging learning tasks. However, it also signals the necessity for deeper insights into the complex interplay among topological attributes influencing NN performance. By shedding light on the advantages and limitations of complex topologies, this study provides valuable guidance for practitioners and paves the way for future endeavors to design more efficient and adaptable neural architectures across various applications.

1. Introduction

Modern neural architectures are widely believed to draw significant design inspiration from biological neuronal networks. The artificial neuron, the fundamental functional unit of neural networks (NNs), is based on the McCulloch–Pitts unit (Fitch, 1944), sharing conceptual similarities with its biological counterpart. Additionally, state-of-the-art convolutional NNs incorporate several operations directly inspired by the mammalian primary visual cortex, such as nonlinear transduction, divisive normalization, and maximum-based pooling of inputs. However, these architectures may be among the few examples where the evolutionary structural and functional properties of neuronal systems have been genuinely relevant for NN design. Indeed, the topology of biological connectomes has not yet been translated into deep learning model engineering.

Due to the ease of implementation and deployment, widely-used neural architectures predominantly feature a regular structure resembling a sequence of functional blocks (e.g., neuronal layers). The underlying multipartite graph of a multilayer perceptron (MLP) is typically controlled by two hyperparameters that define its basic topological properties: depth and width. Only recently have computer vision engineers transitioned from chain-like structures (Simonyan & Zisserman, 2014) to more elaborate connectivity patterns (He, Zhang, Ren, & Sun, 2016; Huang, Liu, Van Der Maaten, & Weinberger, 2017; Xie, Kirillov, Girshick, & He, 2019) (e.g., skip connections, complete graphs). Nevertheless, biological neuronal networks display much richer and less templated wirings at both the micro- and macro-scale (Fornito, Zalesky, & Breakspear, 2013; Xiao, Chen, & Bogdan, 2021; Yang, Sala, & Bogdan, 2021; Yin et al., 2020). For example, considering synaptic connections between individual neurons, the *C. elegans* nematode features a hierarchical modular (Bassett et al., 2010) connectome, wherein hubs with high betweenness centrality are efficiently interconnected (Barthelemy, 2004; Towlson, Vértes, Ahnert, Schafer, & Bullmore, 2013). Moreover, the strength distribution of the adult Drosophila central brain closely follows a power law with an exponential cutoff (Scheffer et al., 2020).

As a result, the relationship between the graph structure of a NN and its predictive abilities remains unclear. In the literature, there is evidence that complex networks can be advantageous in terms of predictive accuracy and parameter efficiency (Kaviani & Sohn, 2021).

* Corresponding author.
E-mail addresses: tommaso.boccato@uniroma2.it (T. Boccato), matteo.ferrante@uniroma2.it (M. Ferrante), duggento@med.uniroma2.it (A. Duggento), toschi@med.uniroma2.it (N. Toschi).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.neunet.2023.12.012
Received 13 April 2023; Received in revised form 7 December 2023; Accepted 7 December 2023
Available online 9 December 2023

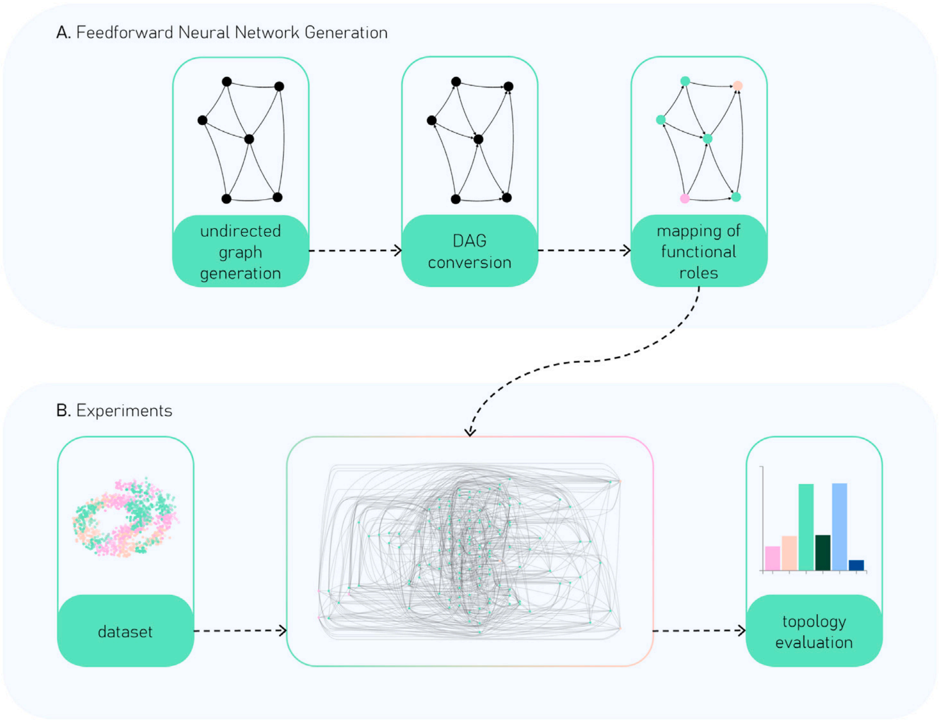

A. Feedforward Neural Network Generation
undirected graph generation
DAG conversion
mapping of functional roles
B. Experiments
dataset
topology evaluation

Fig. 1. Overview of the topology exploration process. **Top**: feedforward neural network (NN) generation. All studied models are constructed using the same three-step procedure. First, we generate an undirected graph with a predetermined degree distribution. Then, we set edge directions and map computational operations to the network nodes. **Bottom**: experiments. For each investigated topology, we sample multiple graphs from the same degree distribution. The corresponding NNs are trained on one of the benchmark datasets. The resulting test accuracies are collected and stored for subsequent analyses.

However, past attempts to investigate this connection have yielded conflicting results that are difficult to generalize outside the investigated context. The first experiment on complex NNs was performed in 2005 by Simard et al. who trained a randomly rewired MLP on random binary patterns (Simard, Nadeau, & Kröger, 2005). Nearly a decade later, Erkaymaz and his collaborators employed the same experimental setup on various real-life problems (Erkaymaz & Ozer, 2016; Erkaymaz, Ozer, & Perc, 2017; Erkaymaz, Özer, & Yumuşak, 2012, 2014) (e.g., diabetes diagnosis, performance prediction of solar air collectors). The best-performing models featured a number of rewirings consistent with the small-world regime. However, all assessed topologies were constrained by MLP-random interpolation. In Annunziato, Bertini, De Felice, and Pizzuti (2007), an MLP and a NN generated following the Barabási–Albert (BA) procedure were compared on a chemical process modeling problem. Both models were trained with an evolutionary algorithm, but the MLP achieved a lower RMSE. The *learning matrix* (Monteiro et al., 2016), a sequential algorithm for the forward/backward pass of arbitrary directed acyclic graphs (DAGs), enabled the evaluation of several well-known complex networks on classification (Monteiro et al., 2016) and regression (Platt, Yang, & Silva Neto, 2019) tasks. The experiments included random and small-world networks, two topologies based on "preferential attachment", a complete graph and a *C. elegans* subnetwork (Dunn, Lockery, Pierce-Shimomura, & Conery, 2004). Nevertheless, the learning matrix's time complexity limited the network sizes (i.e., 26 nodes) and, for each task, a different winning topology emerged, including the MLP. Also Stier et al. successfully trained BA- and WS-based (Watts–Strogatz) NNs with backpropagation (Stier & Granitzer, 2019) on the MNIST classification task (Lecun, Bottou, Bengio, & Haffner, 1998) by placing the generated networks between two fully-connected layers. While this design choice was made in order to adapt the architecture to the dimensionalities of the input/output, it may represent a confound when disentangling the contributions of the different network modules to the overall classification performance. Some recent works have instead focused on multipartite sparse graphs (Mocanu et al., 2018; You, Leskovec, He, & Xie, 2020). While these architectures outperformed the complete baselines, their topological complexity was entirely encoded within the connections between adjacent layers. Another area of research that explores NNs characterized by complex graphs is the Lottery Ticket Hypothesis (LTH) (Frankle & Carbin, 2018). The LTH posits that deep NNs contain subnetworks, often referred to as "winning tickets", with optimized initial weights. When trained in isolation, these subnetworks can achieve high performance on specific tasks. However, it is important to note that these subnetworks are restricted to the "mother" architectures, which typically consist of multipartite graphs or chain-like macro-scale networks.

We propose the hypothesis that, given the same number of nodes (i.e., neurons) and edges (i.e., parameters), a complex NN might exhibit superior predictive abilities compared to classical, more regularly structured MLPs. Unlike previous studies, we conduct a systematic exploration (of which we have reported an overview in Fig. 1) of random, scale-free, and small-world graphs (Fig. 2) on synthetic classification tasks,[2] with particular emphasis on the following:

- **Network size.** The defining properties of a complex topology often emerge in large-scale networks. For example, the second moment of a power-law degree distribution diverges only in the $N \to \infty$ limit (Barabási, 2016), where N is the network size.[3] The networks in Monteiro et al. (2016), Platt et al. (2019) have 15 and 26 nodes, respectively. We trained models with up to 128 neurons.
- **Dataset size.** The *estimation error* achieved by a predictor depends on the training set size: the greater the number of samples, the lower the error (Shalev-Shwartz & Ben-David, 2014). Except

[2] The source code for our experiments is available at https://github.com/BoCtrl-C.

[3] The proposition holds when the degree exponent is smaller than 3.

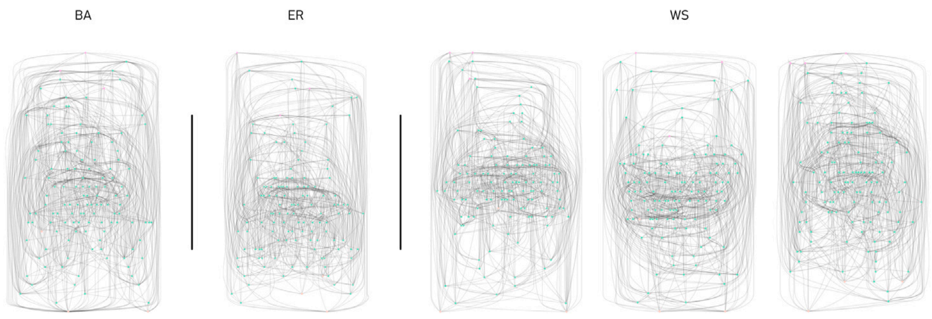


Fig. 2. Example feedforward NNs (128 neurons, 732 synaptic connections) based on complex topologies: scale-free (BA), random (ER), and small-world (WS). All graphs are directed and acyclic. Information flows from top to bottom. Input, hidden, and output units are denoted in dark pink, green, and light pink, respectively. Since networks are defined at the micro-scale, hidden and output nodes implement weighted sums over the incoming edges. In the hidden units, the computational operation is followed by an activation function. The activations of nodes located on the same horizontal layer can be computed in parallel.

for studies based on multipartite graphs, all previous research works in a small-data regime. Our synthetic datasets are three times larger than those used before.

- **Hyperparameter optimization.** Learning rate and batch size are crucial in minimizing the loss function. Monteiro et al. (2016) is the only one that considers finding the optimal learning rate. The role of batch size has never been investigated. Each DAG, however, could be characterized by its optimal combination of hyperparameters. Hence, we optimized the learning rate and batch size for each topology.

Further, we present a series of supplementary experiments aimed at exploring the suitability of non-standard topologies and the applicability of our results to real-world data.

2. Theory

In this section, we briefly report on the network science theory behind graph generators. These graph models are involved in generating the NNs employed in our investigations, as discussed in Section 3.

Erdős–Rényi (ER). An ER graph (Erdős, Rényi, et al., 1960), or *random network*, is uniformly sampled from the set of all graphs with N nodes and L edges. For $N \gg \langle k \rangle$, the degree distribution of a random graph is well approximated by a Poisson distribution: $p_k = e^{-\langle k \rangle}\frac{\langle k \rangle^k}{k!}$; k and $\langle k \rangle$ represent node degree and average degree, respectively.

Watts–Strogatz (WS). The WS generator (Watts & Strogatz, 1998) aims to create graphs that exhibit both high clustering and the *small-world* property; this is achieved by interpolating *lattices* with random networks. The generation starts from a ring in which nodes are connected to their immediate neighbors, and the links are then randomly rewired with probability p.

Barabási–Albert (BA). The well-known BA model (Albert & Barabási, 2002) can be used to generate networks characterized by the $p_k \propto k^{-3}$ *scale-free* degree distribution. Given that the model is inspired by the growth of real networks, the generative procedure iteratively attaches nodes with m stubs to a graph that evolves from an initial star of $m + 1$ nodes. Node additions respond to the preferential attachment mechanism: the probability that a stub reaches a node is proportional to the degree of the latter.

Multilayer Perceptron (MLP). The networks underlying MLPs are called multipartite graphs. In a multipartite graph (i.e., a sequence of bipartite graphs) nodes are partitioned into layers, and each layer can only be connected with the adjacent ones; no intra-layer link is allowed. Additionally, inter-layer connections have to form *bicliques* (i.e., fully-connected bipartite graphs).

We have also reported a comprehensive description of the Stochastic Block Model (SBM) in Appendix C.

3. Methods

The following sections present our methodology. Section 3.1 describes how our synthetic benchmark datasets are constructed. In Section 3.2, we provide details on the proposed NN generation pipeline. Finally, Section 3.3 details out the experimental protocols.

3.1. Datasets

The foundation of the datasets developed, as displayed in Fig. 3, is established by the manifold learning generators[4] provided by the `scikit-learn` machine learning (ML) library (Pedregosa et al., 2011). To modify the generators for classification purposes, 3D points sampled from one of the available curves (*s curve* and *swiss roll*) are segmented into `n_classes` $\times$ `n_reps` portions based on their univariate position relative to the primary dimension of the manifold samples. As the term implies, `n_classes` refers to the number of classes involved in the considered classification. Each segment is then arbitrarily allocated to a class, maintaining task balance (i.e., precisely `n_reps` segments have the same label). We define `n_reps` as the task *difficulty*. An additional aspect of our datasets is the standard deviation σ of the Gaussian noise that can be added to the points. The generation procedure is finalized with a min–max normalization.

3.2. Feedforward neural networks

All trainable models are produced following the same 3-step procedure and, in the context of a specific experiment, share N and L. Consequently, NNs exhibit identical density and parameter counts.

Undirected Graph Generation. The initial step in creating a NN involves sampling an undirected graph using the generators detailed in Section 2. Once N and L are established, all models exhibit a single parameter configuration compatible with the required density.[5] The WS generator is the sole exception: the probability p is allowed to vary between 0 and 1. If the generator is limited to sample networks with a number of links from a finite set (e.g., $L = m + (N - m - 1)m$ according to the BA model), we first generate a graph with slightly higher density than the target before randomly eliminating excess edges. After obtaining the graph, we confirm the existence of a single connected component.

Directed Acyclic Graph (DAG) Conversion. Before performing any calculations, the direction for information propagation through the network links must be determined; this is accomplished by randomly assigning, without replacement, an integer index from $\{1, \dots, N\}$ to the

[4] https://scikit-learn.org/stable/datasets/sample_generators.html

[5] This statement is accurate if the number of MLP layers is predetermined.

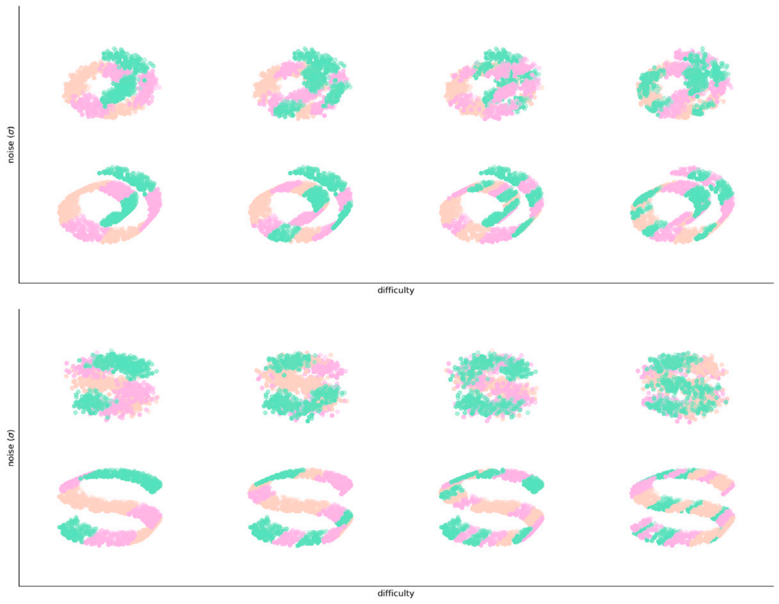


Fig. 3. Benchmark classification datasets. **Top**: the *swiss roll*. **Bottom**: the *s curve*. Each dataset is composed of 3D points divided into multiple segments. Classes are color-coded. Datasets differ in terms of difficulty (x axis) and noise (y axis).

network nodes. It can be shown that the directed graph obtained by setting the direction of each edge from the node with a lower index to the node with a higher index is free of cycles (Bondy & Murty, 1976). However, this conversion results in an unpredictable number of sources and sinks. Since classification tasks typically involve a pre-defined number of input features and output classes, it is necessary to resolve such network-task discrepancies. To address this issue, we developed a straightforward heuristic capable of adjusting DAGs without altering the underlying undirected graphs.

Mapping of Functional Roles. The last step of the presented procedure consists in mapping computational operations to the DAG nodes. Working at the micro-scale (i.e., connections between single neurons), the operations allowed are two. Source nodes implement constant functions; their role, indeed, is to feed the network with the initial conditions for computations. Hidden and sink nodes, instead, perform a weighted sum over the incoming edges, followed by an activation function:

$$a_v = \sigma\left(\sum_u w_{uv} a_u + b\right) \tag{1}$$

where a_v is the activation of node v, σ denotes the activation function[6] (SELU (Klambauer, Unterthiner, Mayr, & Hochreiter, 2017) for hidden nodes and the identity function for sinks), u represents the generic predecessor of v, w_{uv} is the weight associated with edge (u, v) and b the bias. In order to implement the map of functional roles, we made use of the 4Ward library,[7] (Boccato, Ferrante, Duggento, & Toschi, 2023) developed for the purpose. Starting from a DAG, the package returns a working NN deployable as a PyTorch `Module`.

3.3. Experiments

In this section, we outline the experimental protocols designed to evaluate the performance of the diverse graph topologies under scrutiny. In the first paragraphs, we delve into our core experiment — the exploration conducted on synthetic datasets (Protocol 1). Then, we detail our approach to assessing the robustness of the models, previously trained in the aforementioned setup, against node removal (Protocol 2). Shifting focus, in the last paragraphs we discuss our investigation into the influence of network size and density on the overall model performance (Protocol 3) and present our study on the feasibility of employing the investigated computational graphs with real-world data (Protocol 4).

Dataset Partitioning. Each generated dataset is randomly divided into 3 non-overlapping subsets: the train, validation and test splits. All model trainings are performed over the train split while the validation split is exploited in validation epochs and hyperparameter optimization. Test samples, instead, are accessed only in the evaluation of the final models.

Model Training. Models are trained by minimizing cross entropy with the Adam (Kingma & Ba, 2015) optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$). A scheduler reduces the learning rate by a factor of 0.5 if no improvement is seen on the validation loss for 10 epochs. The training procedure ends when learning stagnates (w.r.t. the validation loss) for 15 epochs, and the model weights corresponding to the epoch in which the minimum validation loss has been achieved are saved.

Hyperparameter Optimization. Hyperparameters are optimized through a grid search over a predefined 2D space (i.e., learning rate/batch size). We generate networks of the same topological family starting from 5 different random seeds. In the MLP case, models differ only in the weight initialization. For each parameter pair, the 5 models are trained accordingly, and the resulting best validation losses are collected. Then, the learning rate and batch size that minimize the median validation loss computed across the generation seeds are selected as the optimal hyperparameters of the considered graph family.

Topology Evaluation. Once the optimal learning rate and batch size are found, we train 15 new models characterized by the considered topology and compute mean classification accuracy and standard deviation on the dataset test split. The procedure is repeated for each investigated graph family and a Kruskal–Wallis (H-test) (Kruskal & Wallis, 1952) is performed in order to test the null hypothesis that the medians of all accuracy populations are equal. If the null hypothesis is

[6] Depending on the context, we use the same σ notation for both the standard deviation of the dataset noise and the activation function.

[7] https://github.com/BoCtrl-C/forward

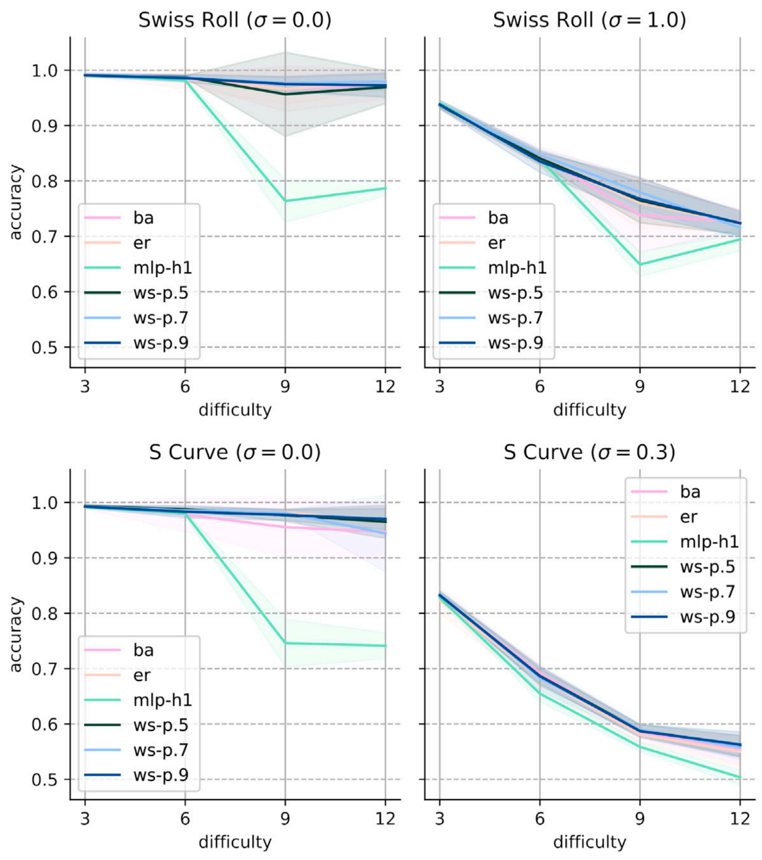


Fig. 4. Mean test accuracy as a function of the task difficulty. Confidence intervals (± standard deviation) are reported as well. Different subplots correspond to different datasets. Each curve denotes the trend of a specific network topology.

rejected, a Mann–Whitney (U-test) (Mann & Whitney, 1947) post hoc analysis follows.

Robustness Analysis. We use the final trained models in a *graph damage* study to investigate their *functional* robustness (accuracy vs. fraction of removed nodes). The *topological* robustness (giant component vs. fraction of removed nodes) is already well-studied in network science. We randomly remove a fixed fraction of nodes, f, from a neural network and compute the accuracy achieved by the resulting model on the test dataset. Practically, node removal is implemented using PyTorch's `Dropout`,[8] which zeroes some network activations by sampling from i.i.d. Bernoulli distributions. As each batch element is associated with specific random variables, activations produced by different dataset samples are processed by differently pruned neural networks. Therefore, the figure of interest is averaged over the dataset and the 15 generation seeds. In a typical topological analysis, when $f = 0$, the giant components of all tested graphs have the same size (i.e., N). We adopt this convention in our experimental setup by replacing test accuracy with *accuracy gain*: $\mathcal{A}(f)$. The metric is defined as the ratio between the accuracy obtained by a pruned network and the accuracy obtained by the original one (i.e., $f = 0$). An accuracy gain < 1 indicates a decline in model performance. Consequently, the figure of merit for our analysis is the mean accuracy gain, with the expectation taken over the generation seeds.

The Role of Size and Density. Size and density are two of the most crucial attributes in a network. In order to explore the influence of these properties on our results, in an additional set of experiments we allow N and L to vary. This adjustment allows us to reveal the impact of these two properties on the performance of the models. For clarification, in this context, the size of a network is defined as the number of neurons it contains. Density, on the other hand, refers to the ratio between the actual number of edges and the maximum number of edges an equivalent undirected network with the same N can have: $\rho = \frac{2L}{N(N-1)}$. As a result, here computational graphs are generated based on a predetermined size/density grid; NNs are then trained and validated using the same synthetic datasets as above, focusing on the four with the highest level of classification difficulty.

Real-World Data. Synthetic data is valuable for characterizing a model's behavior under varying controllable parameters. However, it is equally pivotal to assess whether the results obtained from the investigated architecture can translate to real-world scenarios. To achieve this goal, we conduct additional experiments using the same "fair comparison" framework as outlined in Protocol 1. This framework involves conducting hyperparameter optimization and topology evaluation under consistent conditions, with the same values for N and L. We perform these experiments on six datasets sourced from the UCI suite.[9] Specifically, we handpicked the top-6 classification datasets with fewer than 10 numerical attributes, based on their popularity (i.e., number of views). The resulting list, arranged in descending order, comprises: Iris (Fisher, 1988), Glass Identification (German, 1987), Ecoli (Nakai, 1996), Rice (Rice (Cammeo and Osmancik), 2019), Breast Cancer Wisconsin Wolberg (1992), and Haberman's Survival (Haberman, 1999).

4. Results

The first results presented have been obtained by following Protocol 1, outlined in Section 3, and using the specified topologies (i.e., BA,

[8] https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html

[9] https://archive.ics.uci.edu/datasets

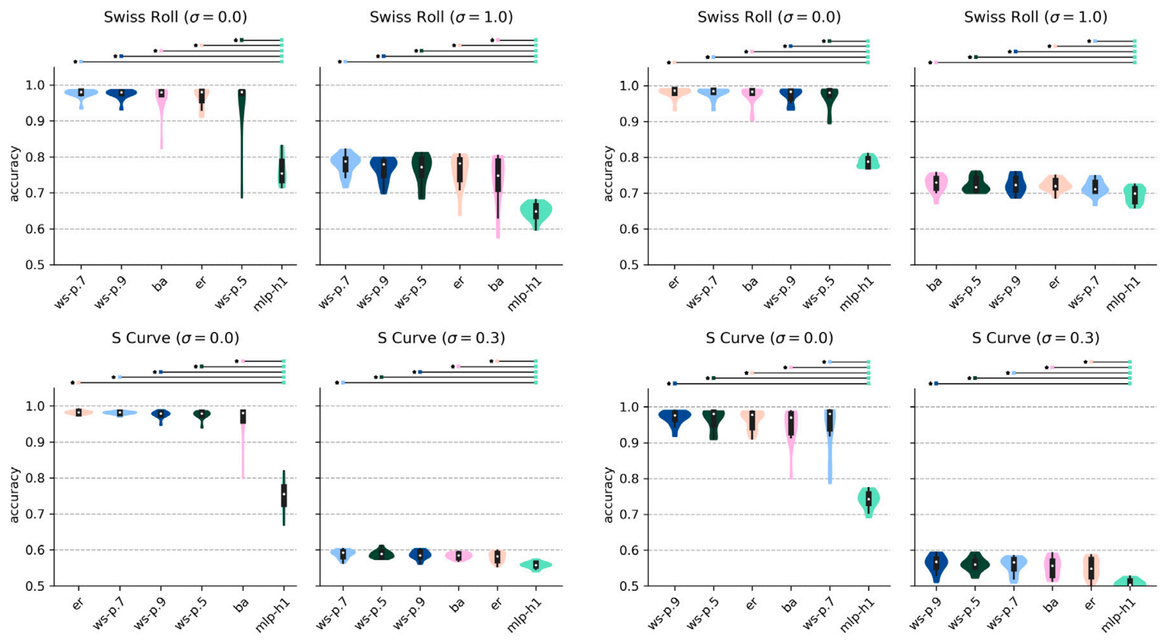


Fig. 5. Accuracy distributions (test split) at the highest difficulty levels. **Left**: difficulty = 9. **Right**: difficulty = 12. Each violin corresponds to a specific network topology and is represented by a consistent color across all plots (following the color scheme from Fig. 4). Statistical annotations appear above the plots, with each segment indicating a significant difference between two medians. Violins in the plots are sorted by mean test accuracy, from left to right in decreasing order.

ER, MLP and WS — further topologies are presented in Appendices B and C) and synthetic datasets. We set `n_classes` = 3 and `n_reps` $\in \{3, 6, 9, 12\}$; for the *swiss roll* dataset, $\sigma \in 0.0, 1.0$, while for the *s curve*, $\sigma \in \{0.0, 0.3\}$. The train, validation, and test split sizes were 1350, 675, and 675, respectively. Given that in a 1-hidden layer MLP (`h1` notation) the number of synaptic connections depends solely on N (i.e., $L = 3 \times H + H \times 3$, with $H = N - 3 - 3$), we chose an MLP with 128 neurons as a reference model and calculated the hyperparameters for the complex networks to achieve graphs with $L = 732$ edges.[10] The additional degree of freedom in the WS generator enabled us to separate the small-world topology into three distinct graph families: `p.5` ($p = 0.5$), `p.7` ($p = 0.7$), and `p.9` ($p = 0.9$). The hyperparameter optimization searched for learning rates in {0.03, 0.01, 0.003, 0.001} and batch sizes in {32, 64}.

Fig. 4 displays the mean test accuracy achieved by each group of models as a function of task difficulty. All manifolds, noise levels, and difficulties are represented. Excluding difficulty level 9 in the *swiss roll* dataset, the accuracy curves exhibit a clear decreasing trend. Specifically, as the difficulty increases, the performance of the MLPs degrades more rapidly than that of complex networks. Confidence intervals, on the other hand, are wider in the high-difficulty plot regions. As expected, noisy tasks were more challenging to learn.

In Fig. 5, the results obtained by the models for the two highest levels of task difficulty are shown in detail. The H-test null hypothesis is rejected for all experiments, and the U-test statistical annotations are displayed. Regardless of the scenario considered, a complex topology consistently holds the top spot in the mean accuracy ranking. MLPs, in contrast, are always the worst-performing models. Moreover, the MLP performance differs significantly from that of the complex networks, in a statistical sense.

Fig. 6 presents the results of the robustness analysis (Protocol 2). We investigated $f \in \{0.0, 0.1, \dots, 0.5\}$ and removed nodes from the models trained on the datasets characterized by the lowest level of difficulty. On these tasks, indeed, all models behave approximately the same (see Fig. 4), hinting at a fair comparison. Unsurprisingly, node removal has the same effect on all topologies: the accuracy gain decreases as f increases. MLPs, however, show enhanced robustness to random deletions. Confidence intervals of the complex graph families overlap. It is worth noting that the chance level (i.e., accuracy of $1/3$) could be reached by different accuracy gains depending on the task; the best accuracy under $f = 0$, indeed, varies between the manifold/noise pairs.

The results obtained from the experiments conducted to investigate the influence of size and density on the approximation capabilities of complex NNs (Protocol 3) are presented in Fig. 7. It is worth noting that the surfaces displayed, which depict test accuracy points linked to different values for the number of neurons and parameters, exhibit a consistent monotonically increasing pattern for both variables, except for a few outliers. Moreover, in nearly all cases, accuracy saturates as the network size approaches $N = 128$. It is important to clarify that in this series of experiments, each data point represents an average computed over 5 training sessions. Learning rate and batch size were held constant at 0.03 and 64, respectively.

Finally, Table 1 presents the performance of the studied models on the UCI real-world data (Protocol 4). For each dataset-model pair, we provide the average test accuracy, computed over 5 runs, along with the associated standard deviation. It is worth noting that these experiments were conducted following the "fair comparison" setup (refer to Protocol 1, Section 3.3), wherein models trained on the same dataset possess an equal number of neurons (128) and parameters. Across different datasets, the number of parameters varies as it is determined by the number of hidden neurons, which is set as N - *the number of features* - *the number of classes*. The number of edges, L, follows the equations introduced at the beginning of this section. Hyperparameters were optimized as previously described for the experiments in line with Protocol 1. In 4 out of 6 datasets, complex NNs outperformed MLPs, whereas in the remaining cases, MLPs exhibited comparable performance to complex NNs.

5. Discussion

In this paper, the most significant finding is the performance, in terms of accuracy, attained by the architectures built on complex

[10] With this particular dataset hyperparameter selection, the neural network's output will consist of a 3-dimensional vector. It is important to note that the input, which represents the 3D coordinates of a sample point on the curve, must also maintain a dimensionality of 3.

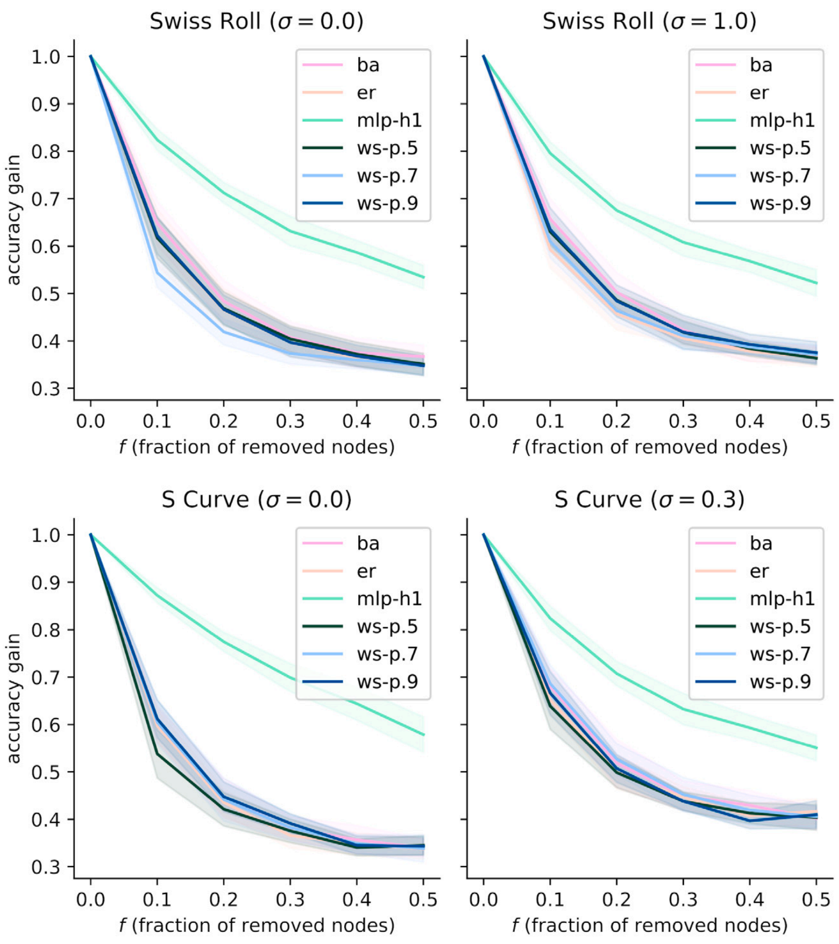


Fig. 6. Robustness analysis. The horizontal axis reports the fraction of removed nodes (i.e., f) while the vertical one the accuracy gain (i.e., $\mathcal{A}(f)$). Each curve refers to a different network topology. Confidence intervals (± standard deviation) are reported.

Table 1
Performance of various graph families on the UCI datasets. For each entry, mean test accuracy and standard deviation are reported. On each row, the best result is highlighted in bold.

Dataset	ba	er	ws-p.5	ws-p.7	ws-p.9	mlp-h1
Breast	**97.31%** (±0.27)	96.82% (±0.27)	97.21% (±0.44)	97.11% (±0.42)	97.21% (±1.25)	96.72% (±0.44)
Ecoli	84.95% (±4.61)	87.72% (±1.50)	81.78% (±3.33)	87.52% (±1.93)	86.53% (±2.49)	**87.92%** (±0.83)
Glass	**69.23%** (±4.21)	63.38% (±5.03)	68.00% (±5.48)	69.23% (±2.88)	62.77% (±9.32)	64.00% (±2.06)
Haberman	70.45% (±1.97)	**72.05%** (±1.90)	72.05% (±1.72)	70.00% (±1.90)	71.14% (±0.62)	69.09% (±0.95)
Iris	**93.04%** (±0.97)	92.61% (±1.94)	92.61% (±1.19)	92.61% (±1.19)	92.17% (±1.94)	92.17% (±1.19)
Rice	92.66% (±0.25)	92.47% (±0.07)	92.41% (±0.34)	92.45% (±0.18)	92.47% (±0.27)	**92.88%** (±0.30)

topologies both in high-difficulty scenarios using synthetic data and in classification problems defined within the UCI dataset suite. In this context, and in light of the statistical tests carried out, the complex models prove to be a solid alternative to MLPs.

Formally justifying the observed phenomenon is challenging. Fortunately, in 2017, Poggio et al. discussed two theorems (Poggio, Mhaskar, Rosasco, Miranda, & Liao, 2017) that guided our explanation. According to the first theorem,[11] a shallow network (e.g., an MLP `h1`) equipped with infinitely differentiable activation functions requires $N = \mathcal{O}(\epsilon^{-n})$ units to approximate a continuous function f of n variables[12] with an approximation error of at most $\epsilon > 0$. This exponential dependency is technically called the *curse of dimensionality*. On the other hand, the second theorem states that if f is compositional and the network presents its same architecture, we can escape the “curse”. It is important to remember that a compositional function is defined as a composition of “local” constituent functions, $h \in \mathcal{H}$ (e.g., $f(x_1, x_2, x_3) = h_2(h_1(x_1, x_2), x_3)$, where $x_1,\ x_2,\ x_3$ are the input variables and $h_1,\ h_2$ the constituent functions). In other words, the structure of a compositional function can be represented by a DAG. In this approximation scenario, the required number of units depends on $N = \mathcal{O}(\sum_h \epsilon^{-n_h})$, where n_h is the input dimensionality of function h. If $\max_h n_h = d$, then $\sum_h \epsilon^{-n_h} \leq \sum_h \epsilon^{-d} = |\mathcal{H}|\epsilon^{-d}$.

The primary advantage of complex networks is their potential to avoid the curse of dimensionality when relevant graphs for the function to be learned are present. Under the assumption that the function linking the *swiss roll* and *s curve* points to the ground truth labels is compositional (intuitively, in non-noisy datasets, each class is a union of various segments), we conjecture that our complex NNs can exploit this compositionality. In the high-difficulty regime, the necessary network size for MLP `h1` to achieve the same accuracy as complex models

[11] We invite the reader to consult Poggio et al. (2017) for a complete formulation of the theorems.

[12] Depending on the context, we use the same f notation for both the fraction of removed nodes and the function to be approximated.

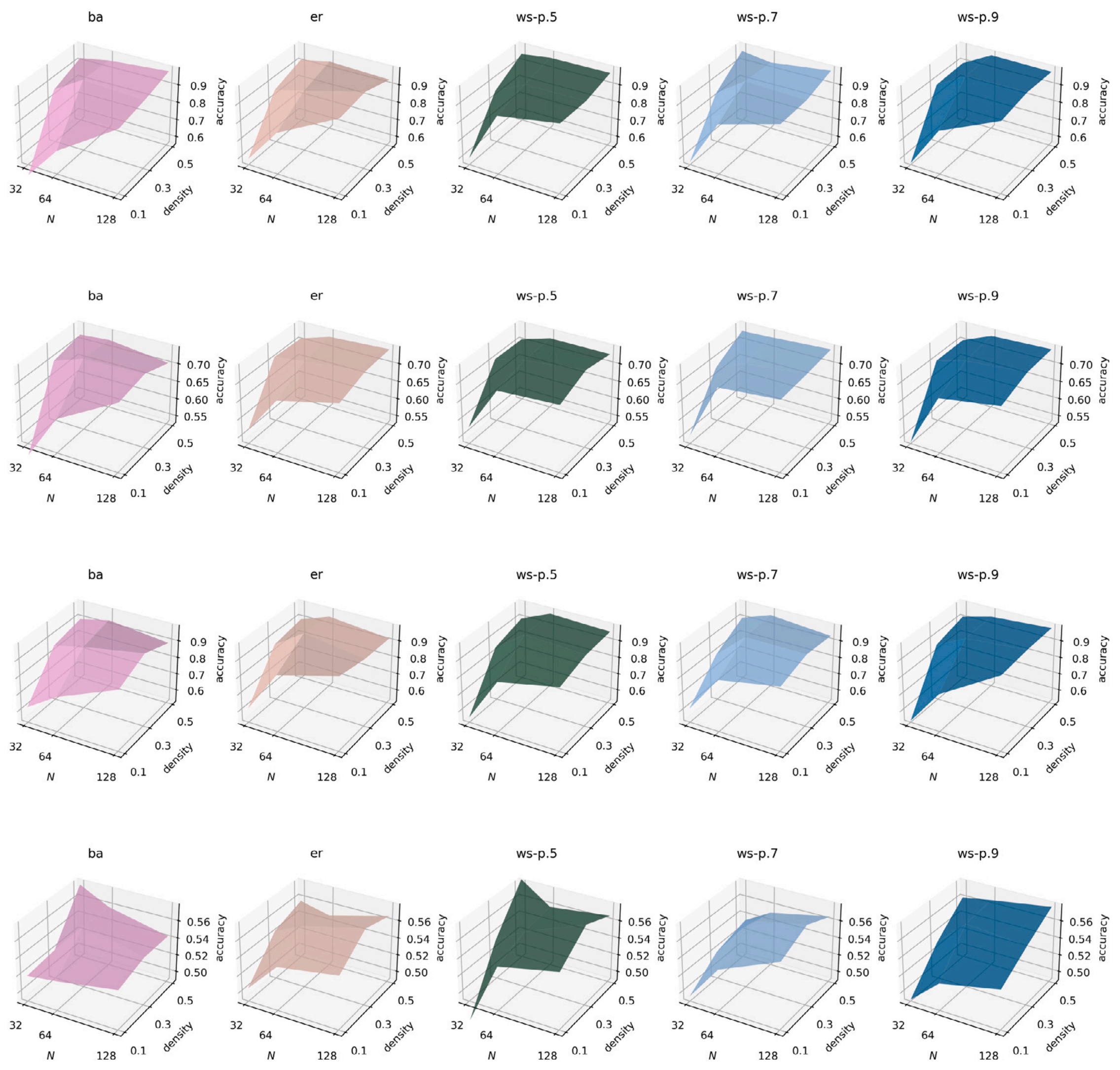


Fig. 7. Mean test accuracy as function of size (N) and density (ρ). From top to bottom: *swiss roll*, noisy *swiss roll*, *s curve* and noisy *s curve*.

likely exceeds the size set for experiments. While one could argue that the datasets employed were compositionally sparse by chance, according to Poggio (2022), all *efficiently computable functions* must be *compositionally sparse* (i.e., their constituent functions have "small" d). Performance differences on noisy datasets are less noticeable, possibly due to the minimal overlap between the functions to be approximated and the studied topologies. Notably, our setup does not precisely match the theorem formulations in Poggio et al. (2017) (e.g., SELUs are not infinitely differentiable), but Poggio et al. argue that the hypotheses can likely be relaxed. No statistically significant differences emerged between the complex graph families from the results of Section 4. Various explanations exist for this outcome: all tested topologies could be complex enough to include relevant subgraphs of the target f functions; the random DAG conversion heuristic might have perturbed hidden topological properties of the original undirected networks; or the degree distribution of a network may not be the most relevant topological feature in a model's approximation capabilities. Additionally, the results in Table 1 demonstrate that the optimal topology of a NN is task-specific. This suggests the need for future advancements in the state-of-the-art of neural architecture search, potentially formulating the creation of computational graphs through data-driven approaches.

The higher accuracy in complex networks, however, comes with trade-offs. Although the methodology in Boccato et al. (2023) improves the scalability of complex NNs and enables experimentation with arbitrary DAGs, it is important to note that 1-hidden layer MLPs typically have faster forward pass computation. In these models, the forward pass requires only two matrix multiplications, whereas, in NNs built using 4Ward, the number of operations depends on the DAG *height*. However, we believe that, in the future, the computational efficiency of tools like 4Ward will be enhanced through the integration of ML frameworks optimized for sparse tensor processing (Nikdan, Pegolotti, Iofinova, Kurtic, & Alistarh, 2023) and specialized hardware (Le Gallo et al., 2022). Moreover, the analyses in Fig. 6 demonstrate the MLPs' superiority in a graph damage scenario. We speculate that the hidden units in an MLP `h1` contribute equally to the approximation of the target function. In contrast, the ability of complex networks to exploit the compositionality of the function to be learned might lead to high specialization of some hidden units.

6. Conclusions

Our study explores the impact of network topology on the performance of artificial NNs, juxtaposing conventional MLPs against an array of models based on complex topologies. Through comprehensive experiments conducted on synthetic datasets, we observed a distinct performance superiority of complex NNs, particularly excelling in addressing high-difficulty learning scenarios, surpassing MLPs in accuracy. This finding was further reinforced by experiments conducted on real-world datasets from the UCI suite, wherein complex networks exhibited enhanced performance in most cases, while in certain instances, MLPs displayed comparable performance.

However, the observed performance in complex NNs was counterbalanced by increased computational demands and reduced robustness to graph damage when compared to MLPs. The intricate relationship observed between topological attributes and model performance underscores a multifaceted interplay, indicating the complex nature of their impact on NN efficiency.

The findings discussed offer guidance for practitioners and researchers, emphasizing the need to consider both the advantages and limitations of employing complex network structures when devising adaptable neural architectures across diverse applications. Furthermore, this study lays the groundwork for future investigations aimed at uncovering optimal topological features, refining construction methodologies, and enhancing the understanding of intricate relationships among topological attributes to engineer more efficient and robust neural architectures.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The source code for our experiments is available at https://github.com/BoCtrl-C.

Acknowledgments

This work is supported and funded by: #NEXTGENERATIONEU (NGEU); the Ministry of University and Research (MUR); the National Recovery and Resilience Plan (NRRP); project MNESYS (PE0000006, to NT) - *A Multiscale integrated approach to the study of the nervous system in health and disease* (DN. 1553 11.10.2022); the MUR-PNRR M4C2I1.3 PE6 project PE00000019 Heal Italia (to NT); the NATIONAL CENTRE FOR HPC, BIG DATA AND QUANTUM COMPUTING, within the spoke *"Multiscale Modeling and Engineering Applications"* (to NT); the European Innovation Council (Project CROSSBRAIN - Grant Agreement 101070908, Project BRAINSTORM - Grant Agreement 101099355); the Horizon 2020 research and innovation Programme (Project EXPERIENCE - Grant Agreement 101017727). Tommaso Boccato is a Ph.D. student enrolled in the National PhD in Artificial Intelligence, XXXVII cycle, course on Health and Life Sciences, organized by Università Campus Bio-Medico di Roma.

Appendix A. Graph attributes

Inspired by the works in Janik and Nowak (2020) and Znaidi et al. (2023), we delved deeper into the role of network topology in the models' approximation capabilities by calculating a total of 27 topological graph attributes for each trained neural network. Our aim was to ascertain if any specific attributes could account for the models' performance. The correlation plots that display the relationship between test accuracy and graph attribute can be found in Figs. A.8 and A.9.

To compute these metrics, we employed the NetworkX library (Hagberg, Schult, & Swart, 2008) when feasible, and devised custom implementations for the remaining attributes. We conducted the experiments using noise-free datasets with the highest difficulty levels.

After analyzing the experimental data, no evident relationship emerged between the attributes and the models' performance. In other words, when examined individually, none of the attributes could account for the achieved accuracies. This outcome implies that the impact of network topology on the approximation capabilities of the models might be more intricate than a straightforward correlation with any single topological attribute. Further investigation is necessary to explore the potential interplay among multiple attributes and their influence on the models' performance.

Appendix B. Hub-based topological orderings

When creating an undirected BA graph, a significant number of hubs emerge alongside numerous low-degree nodes. This raises the question of what happens when, during the DAG conversion, nodes are arranged based on their degree. To investigate this, we have chosen to focus on three key topological orderings: sorting nodes by degree in descending order, sorting nodes by degree in ascending order, and sorting nodes in descending order but starting from the center of the node sequence and extending towards the edges. In other words, the input features can be assigned to the largest hubs, the output logits can be extracted from the largest hubs, or high-degree nodes can be positioned in the middle of the information flow.

For each new topological ordering, we trained 5 models, each with 128 neurons and 732 parameters, on the *swiss roll* and *s curve* datasets characterized by the highest level of difficulty. The hyperparameter search and training procedures followed the same methodology as in all other experiments in this paper. The accuracy distributions resulting from evaluating these models on the respective test splits are depicted in Fig. B.10. Here the group labeled as "ba" represents computational graphs that served as a baseline. These baseline graphs were obtained by converting them through a standard random topological sorting. Regrettably, our experiments did not reveal any statistically significant differences between the various model families. This outcome allows for two distinct interpretations: either the placement of hubs within the topological orderings, which determine the direction of connections, does not significantly impact the models' approximation capabilities, or the NNs employed do not possess a sufficient number of nodes to render hubs a critical factor during the training process. Scaling these experiments in the future would potentially address this ambiguity.

Appendix C. The stochastic block model

The ER, BA, and WS models are among the most well-known tools used for generating undirected networks that exhibit non-trivial degree distributions; however, this is not an exhaustive list. For instance, the Stochastic Block Model (SBM) is another widely recognized generative model for random graphs. It establishes connections between nodes based on their membership in specific communities. The generative algorithm for the SBM requires two key parameters: a partition of the vertex set and a symmetric matrix P, which contains edge probabilities. Each element P_{ij} in the matrix defines the probability of nodes from community i connecting with nodes from community j. When all entries in matrix P are constant, the model reverts to the ER one. Conversely, if both the diagonal and off-diagonal entries are equal, we refer to it as the *planted partition model*. In this case, if we denote the intra-community probability as p and the inter-community probability as q, the model is categorized as *assortative* when $p > q$ and *disassortative* when $p < q$.

Due to the numerous parameters involved in the generation process, we conducted an experiment focused on the planted partition model. The objective was to investigate whether this topology could effectively serve as the basis for a neural network and to evaluate the impact of partition size, intra- and inter-community probabilities on the overall performance of the generated architectures. We set two different partition sizes, namely 4 and 8, with $N = 128$ and $\mathbb{E}[L] = 732$. For each number of communities, we examined two distinct scenarios: the assortative model characterized by the highest possible p, and the disassortative one with the highest possible q, in an effort to explore as diverse configurations as possible. In the first scenario, we can observe

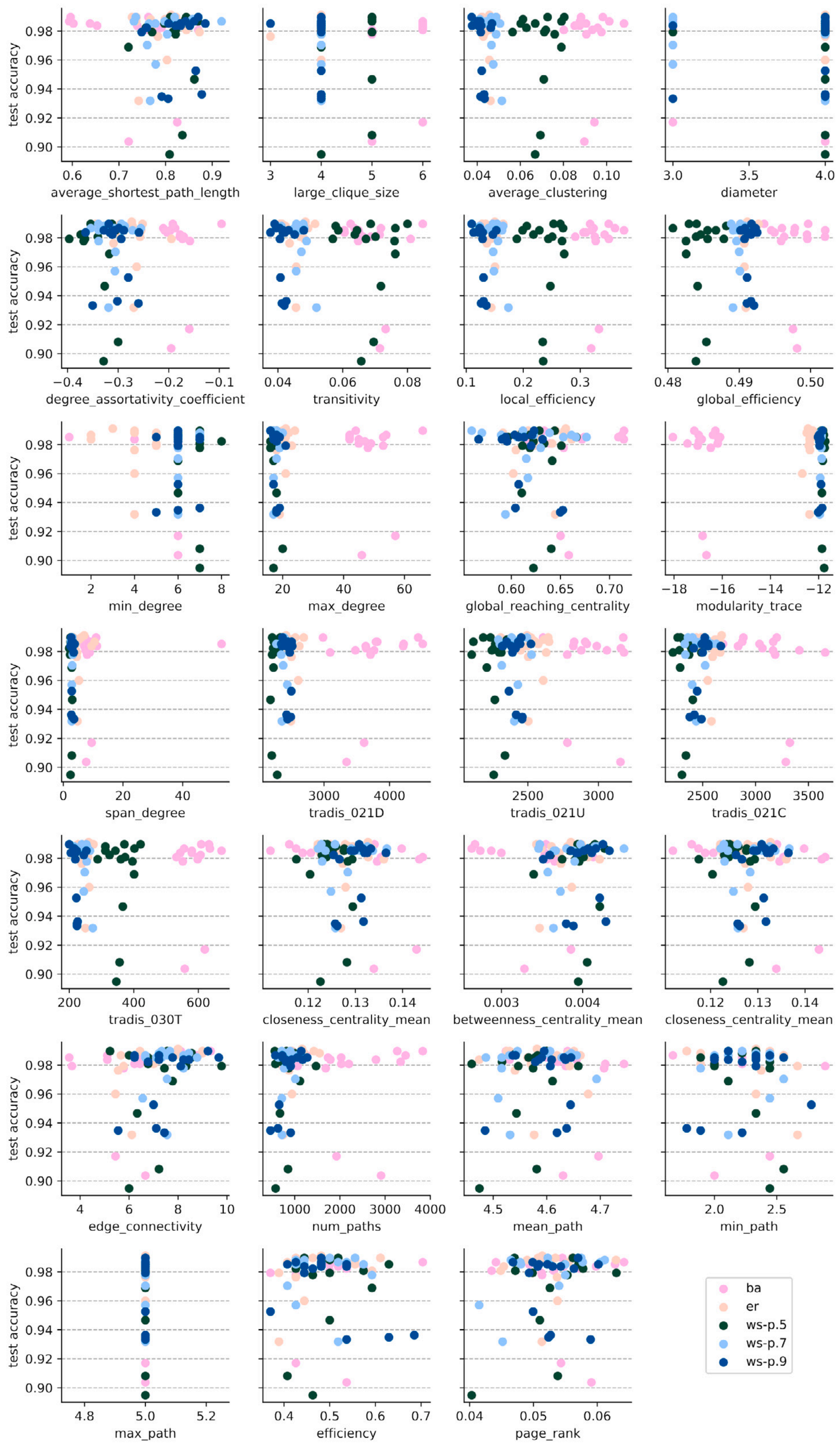


Fig. A.8. Correlation plots (accuracy vs. attribute) computed on the *swiss roll* dataset (`n_reps` = 12, $\sigma = 0.0$). Each network topology is denoted with a different color, which can be found in the legend (last subplot).

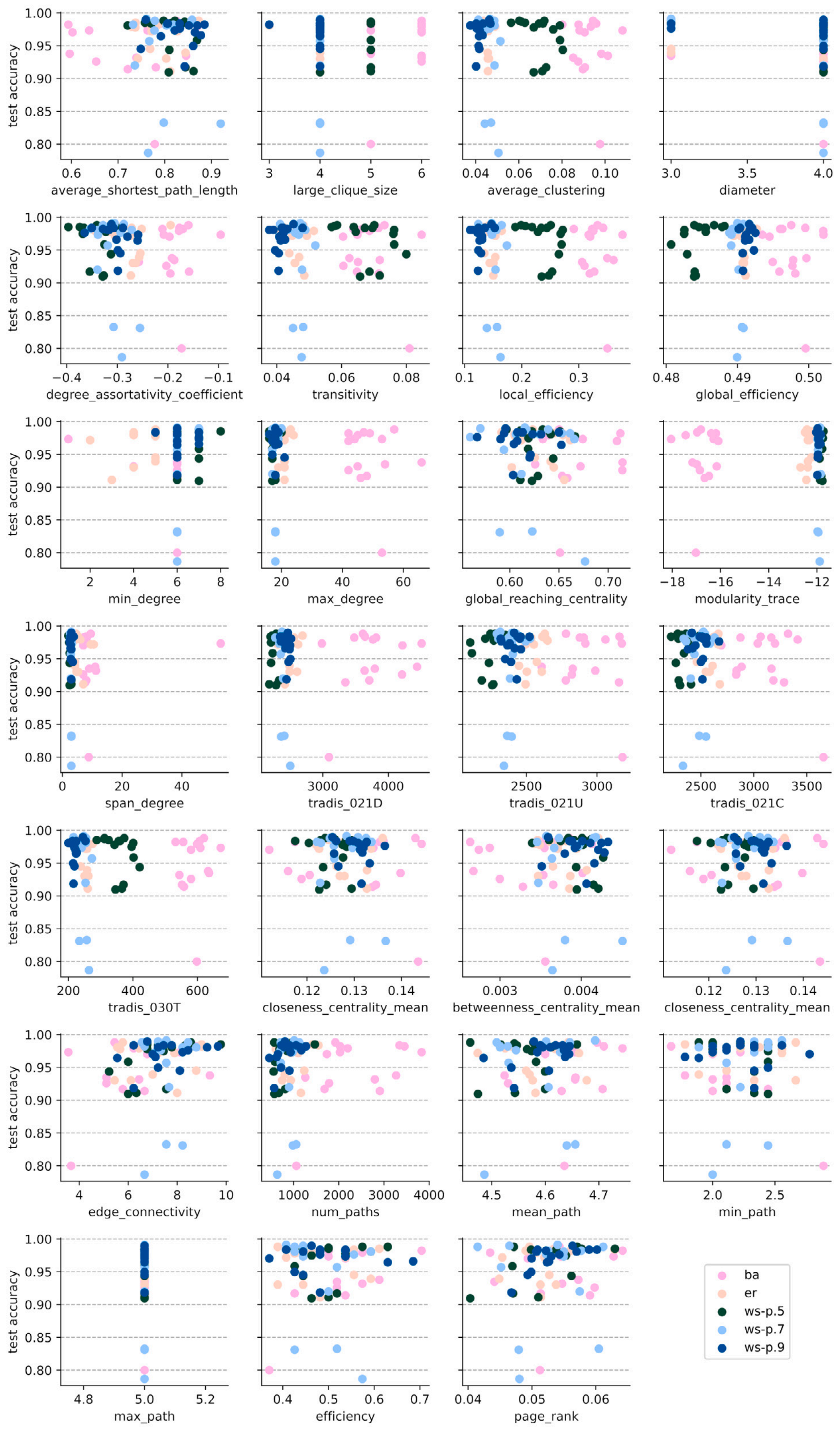


Fig. A.9. Correlation plots (accuracy vs. attribute) computed on the *s curve* dataset (`n_reps` = 12, $\sigma = 0.0$). Each network topology is denoted with a different color, which can be found in the legend (last subplot).

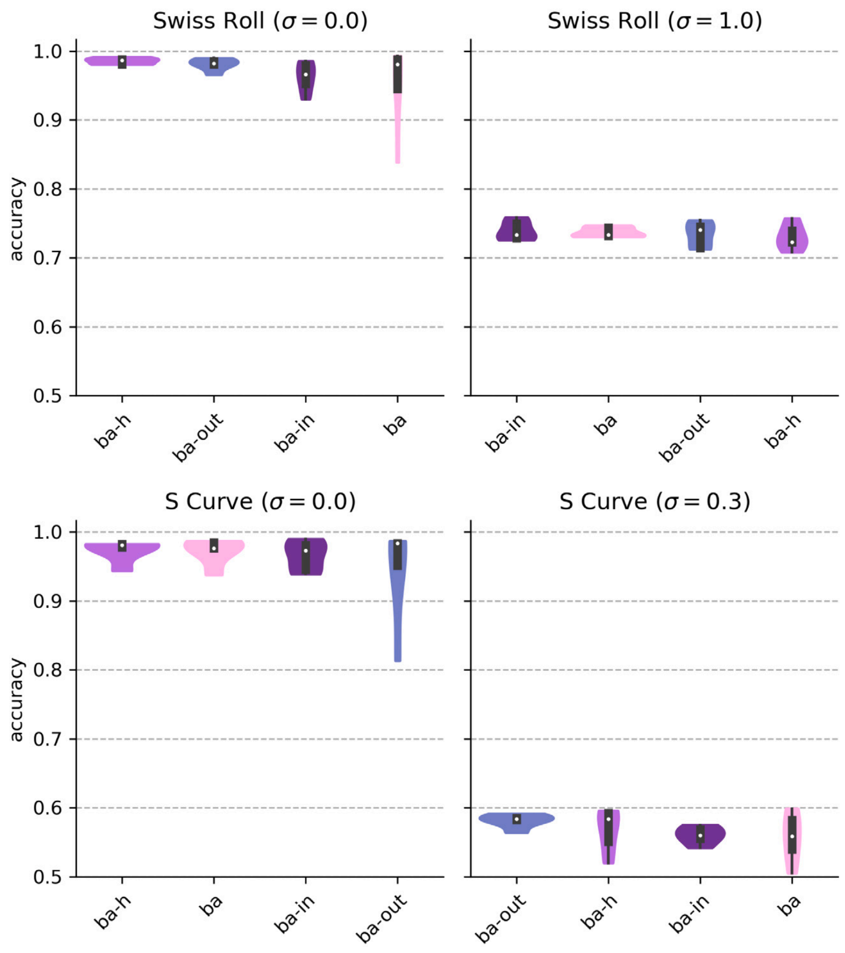


Fig. B.10. Accuracy distributions (test split) for BA NNs generated through different degree-aware topological sortings. Violins in the plots are sorted by mean test accuracy (from left to right).

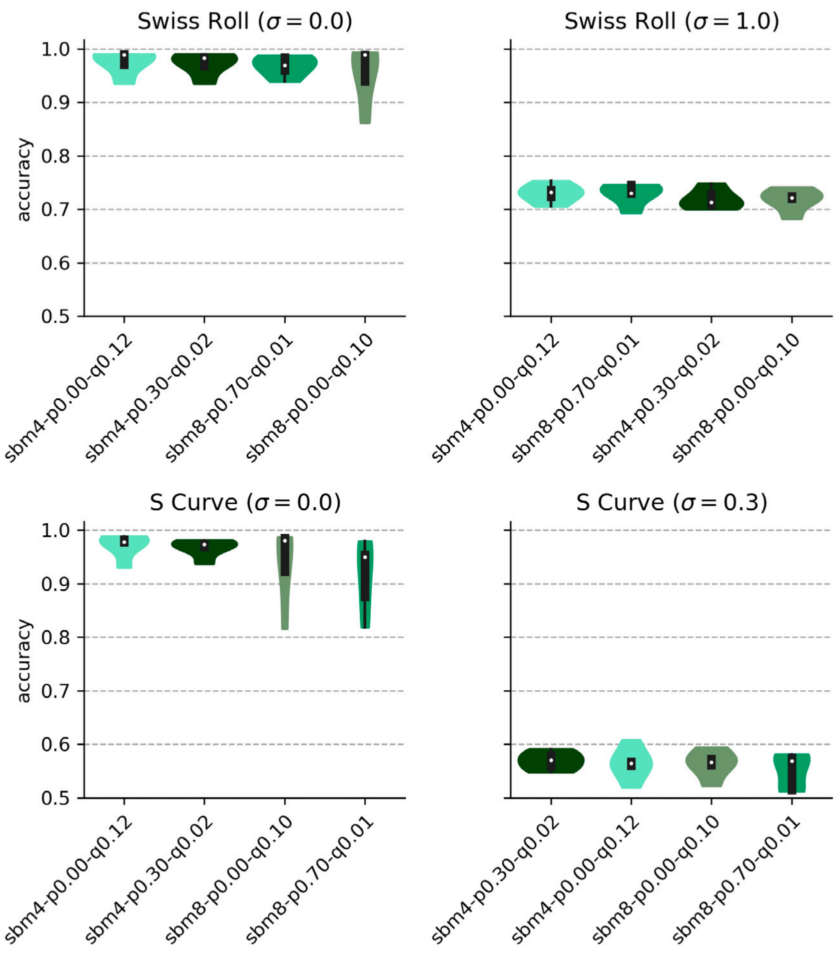


Fig. C.11. Accuracy distributions (test split) for various SBM NNs. Model names follow the following coding: sbm$x$-py-qz where x, y and z represent the number of communities, the intra-community and the inter-community probability, respectively. Violins in the plots are sorted by mean test accuracy (from left to right).

highly dense clusters with sparse inter-cluster connections, while in the second scenario, sparse bipartite graphs within a fully-connected meta-graph. For each configuration, we trained 5 models after conducting a preliminary hyperparameter search. Results are reported in Fig. C.11. While optimization have converged in all experiments, differences in accuracy distributions among the models are not statistically significant. Nevertheless, it is worth noting that the 4-community graphs consistently exhibit the highest average test accuracy across most of the tested tasks.

Appendix D. Glossary

- Artificial Neural Network (ANN)
- Barabási–Albert (BA)
- Directed Acyclic Graph (DAG)
- Erdős–Rényi (ER)
- Lottery Ticket Hypothesis (LTH)
- Multilayer Perceptron (MLP)
- Neural Network (NN)
- Root Mean Square Error (RMSE)
- Watts–Strogatz (WS)

References

Albert, R., & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*(1), 47.

Annunziato, M., Bertini, I., De Felice, M., & Pizzuti, S. (2007). Evolving complex neural networks. In R. Basili, & M. T. Pazienza (Eds.), *Vol. 4733*, *AI*IA 2007: Artificial intelligence and human-oriented computing* (pp. 194–205). Berlin, Heidelberg: Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-540-74782-6_18, Series Title: Lecture Notes in Computer Science, URL http://link.springer.com/10.1007/978-3-540-74782-6_18.

Barabási, A. (2016). *Network science*. Cambridge University Press, URL https://networksciencebook.com/.

Barthelemy, M. (2004). Betweenness centrality in large complex networks. *The European Physical Journal B*, *38*(2), 163–168.

Bassett, D. S., Greenfield, D. L., Meyer-Lindenberg, A., Weinberger, D. R., Moore, S. W., & Bullmore, E. T. (2010). Efficient physical embedding of topologically complex information processing networks in brains and computer circuits. *PLoS Computational Biology*, *6*(4), Article e1000748.

Boccato, T., Ferrante, M., Duggento, A., & Toschi, N. (2023). 4Ward: A relayering strategy for efficient training of arbitrarily complex directed acyclic graphs. *Neurocomputing*, Article 127058. http://dx.doi.org/10.1016/j.neucom.2023.127058, URL https://www.sciencedirect.com/science/article/pii/S0925231223011815.

Bondy, J., & Murty, U. (1976). *Graph theory with applications*. American Elsevier Publishing Company, URL https://books.google.it/books?id=4bwrAAAAYAAJ.

Dunn, N. A., Lockery, S. R., Pierce-Shimomura, J. T., & Conery, J. S. (2004). A neural network model of chemotaxis predicts functions of synaptic connections in the nematode *Caenorhabditis elegans*. *Journal of Computational Neuroscience*, *17*(2), 137–147. http://dx.doi.org/10.1023/B:JCNS.0000037679.42570.d5.

Erdős, P., Rényi, A., et al. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, *5*(1), 17–60.

Erkaymaz, O., & Ozer, M. (2016). Impact of small-world network topology on the conventional artificial neural network for the diagnosis of diabetes. *Chaos, Solitons & Fractals*, *83*, 178–185. http://dx.doi.org/10.1016/j.chaos.2015.11.029, URL https://linkinghub.elsevier.com/retrieve/pii/S096007791500394X.

Erkaymaz, O., Ozer, M., & Perc, M. (2017). Performance of small-world feedforward neural networks for the diagnosis of diabetes. *Applied Mathematics and Computation*, *311*, 22–28. http://dx.doi.org/10.1016/j.amc.2017.05.010, URL https://linkinghub.elsevier.com/retrieve/pii/S0096300317302989.

Erkaymaz, O., Özer, M., & Yumuşak, N. (2012). Performance analysis of a feed-forward artifical neural network with small-world topology. *Procedia Technology*, *1*, 291–296. http://dx.doi.org/10.1016/j.protcy.2012.02.062, URL https://linkinghub.elsevier.com/retrieve/pii/S2212017312000631.

Erkaymaz, O., Özer, M., & Yumuşak, N. (2014). Impact of small-world topology on the performance of a feed-forward artificial neural network based on 2 different real-life problems. (p. 12).

Fisher, R. A. (1988). Iris. http://dx.doi.org/10.24432/C56C76, UCI Machine Learning Repository.

Fitch, F. B. (1944). Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. Bulletin of mathematical biophysics, vol. 5 (1943) 115–133. *Journal of Symbolic Logic*, *9*(2), 49–50. http://dx.doi.org/10.2307/2268029.

Fornito, A., Zalesky, A., & Breakspear, M. (2013). Graph analysis of the human connectome: Promise, progress, and pitfalls. *NeuroImage*, *80*, 426–444. http://dx.doi.org/10.1016/j.neuroimage.2013.04.087, Mapping the Connectome, URL https://www.sciencedirect.com/science/article/pii/S1053811913004345.

Frankle, J., & Carbin, M. (2018). The lottery ticket hypothesis: Training pruned neural networks. CoRR, arXiv:1803.03635, URL http://arxiv.org/abs/1803.03635.

Gallo, M. L., Khaddam-Aljameh, R., Stanisavljevic, M., Vasilopoulos, A., Kersting, B., Dazzi, M., et al. (2022). A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. http://dx.doi.org/10.48550/arXiv.2212.02872, CoRR arXiv:2212.02872.

German, B. (1987). Glass identification. http://dx.doi.org/10.24432/C5WW2P, UCI Machine Learning Repository.

Haberman, S. (1999). Haberman's survival. http://dx.doi.org/10.24432/C5XK51, UCI Machine Learning Repository.

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, & J. Millman (Eds.), *Proceedings of the 7th python in science conference* (pp. 11–15). Pasadena, CA USA.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 770–778). http://dx.doi.org/10.1109/CVPR.2016.90.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2261–2269). http://dx.doi.org/10.1109/CVPR.2017.243.

Janik, R. A., & Nowak, A. (2020). Neural networks on random graphs. CoRR, arXiv:2002.08104, URL https://arxiv.org/abs/2002.08104.

Kaviani, S., & Sohn, I. (2021). Application of complex systems topologies in artificial neural networks optimization: An overview. *Expert Systems with Applications*, *180*, Article 115073. http://dx.doi.org/10.1016/j.eswa.2021.115073, URL https://www.sciencedirect.com/science/article/pii/S0957417421005145.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Y. Bengio, & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, conference track proceedings*. URL http://arxiv.org/abs/1412.6980.

Klambauer, G., Unterthiner, T., Mayr, A., & Hochreiter, S. (2017). Self-normalizing neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Vol. 30*, *Advances in neural information processing systems*. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper_files/paper/2017/file/5d44ee6f2c3f71b73125876103c8f6c4-Paper.pdf.

Kruskal, W. H., & Wallis, W. A. (1952). Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, *47*(260), 583–621, URL http://www.jstor.org/stable/2280779.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. http://dx.doi.org/10.1109/5.726791.

Mann, H. B., & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, *18*(1), 50–60, URL http://www.jstor.org/stable/2236101.

Mocanu, D. C., Mocanu, E., Stone, P., Nguyen, P. H., Gibescu, M., & Liotta, A. (2018). Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, *9*(1), 2383. http://dx.doi.org/10.1038/s41467-018-04316-3, URL http://www.nature.com/articles/s41467-018-04316-3.

Monteiro, R. L. S., Carneiro, T. K. G., Fontoura, J. R. A., da Silva, V. L., Moret, M. A., & Pereira, H. B. d. B. (2016). A model for improving the learning curves of artificial neural networks. *PLOS ONE*, *11*(2), Article e0149874. http://dx.doi.org/10.1371/journal.pone.0149874, URL https://dx.plos.org/10.1371/journal.pone.0149874.

Nakai, K. (1996). Ecoli. http://dx.doi.org/10.24432/C5388M, UCI Machine Learning Repository.

Nikdan, M., Pegolotti, T., Iofinova, E., Kurtic, E., & Alistarh, D. (2023). SparseProp: Efficient sparse backpropagation for faster training of neural networks at the edge. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, & J. Scarlett (Eds.), *Proceedings of machine learning research*: *Vol. 202*, *Proceedings of the 40th international conference on machine learning* (pp. 26215–26227). PMLR, URL https://proceedings.mlr.press/v202/nikdan23a.html.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Platt, G. M., Yang, X.-S., & Silva Neto, A. J. (Eds.), (2019). *Computational intelligence, optimization and inverse problems with applications in engineering*. Cham: Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-96433-1, URL http://link.springer.com/10.1007/978-3-319-96433-1.

Poggio, T. (2022). It is compositional sparsity: a framework for ML. (p. 9).

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., & Liao, Q. (2017). Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 1–17. http://dx.doi.org/10.1007/s11633-017-1054-2, URL http://link.springer.com/article/10.1007/s11633-017-1054-2?wt_mc=Internal.Event.1.SEM.ArticleAuthorOnlineFirst.

Rice (Cammeo and Osmancik). (2019). http://dx.doi.org/10.24432/C5MW4Z, UCI Machine Learning Repository.

Scheffer, L. K., Xu, C. S., Januszewski, M., Lu, Z., Takemura, S.-y., Hayworth, K. J., et al. (2020). A connectome and analysis of the adult Drosophila central brain. *eLife*, *9*, Article e57443. http://dx.doi.org/10.7554/eLife.57443, URL https://elifesciences.org/articles/57443.

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: from theory to algorithms*. Cambridge University Press, http://dx.doi.org/10.1017/CBO9781107298019.

Simard, D., Nadeau, L., & Kröger, H. (2005). Fastest learning in small-world neural networks. *Physics Letters. A*, *336*(1), 8–15. http://dx.doi.org/10.1016/j.physleta.2004.12.078, URL https://linkinghub.elsevier.com/retrieve/pii/S0375960105000022.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv e-prints, arXiv:1409.1556.

Stier, J., & Granitzer, M. (2019). Structural analysis of sparse neural networks. *Procedia Computer Science*, *159*, 107–116. http://dx.doi.org/10.1016/j.procs.2019.09.165, Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Conference KES2019, URL https://www.sciencedirect.com/science/article/pii/S1877050919313432.

Towlson, E. K., Vértes, P. E., Ahnert, S. E., Schafer, W. R., & Bullmore, E. T. (2013). The rich club of the C. elegans neuronal connectome. *Journal of Neuroscience*, *33*(15), 6380–6387.

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world'networks. *Nature*, *393*(6684), 440–442.

Wolberg, W. (1992). Breast cancer wisconsin (original). http://dx.doi.org/10.24432/C5HP4Z, UCI Machine Learning Repository.

Xiao, X., Chen, H., & Bogdan, P. (2021). Deciphering the generating rules and functionalities of complex networks. *Scientific Reports*, *11*(1), http://dx.doi.org/10.1038/s41598-021-02203-4, https://par.nsf.gov/biblio/10351662.

Xie, S., Kirillov, A., Girshick, R., & He, K. (2019). Exploring randomly wired neural networks for image recognition. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*.

Yang, R., Sala, F., & Bogdan, P. (2021). Hidden network generating rules from partially observed complex networks. *Communications Physics*, *4*(1), http://dx.doi.org/10.1038/s42005-021-00701-5, https://par.nsf.gov/biblio/10351663.

Yin, C., Xiao, X., Balaban, V., Kandel, M. E., Lee, Y. J., Popescu, G., et al. (2020). Network science characteristics of brain-derived neuronal cultures deciphered from quantitative phase imaging data. *Scientific Reports*, *10*(1), http://dx.doi.org/10.1038/s41598-020-72013-7, https://par.nsf.gov/biblio/10289849.

You, J., Leskovec, J., He, K., & Xie, S. (2020). Graph structure of neural networks. arXiv:2007.06559, [cs, stat], URL http://arxiv.org/abs/2007.06559.

Znaidi, M. R., Sia, J., Ronquist, S., Rajapakse, I., Jonckheere, E., & Bogdan, P. (2023). A unified approach of detecting phase transition in time-varying complex networks. *Scientific Reports*, *13*(1), 17948.