Modularity and multitasking in neuro-memristive reservoir networks

Alon Loeffler, Ruomin Zhu, Joel Hochstetter, Adrian Diaz-Alvarez, Tomonobu Nakayama, James M Shine and Zdenka Kuncic

1 The University of Sydney, School of Physics, Sydney, Australia
2 The University of Sydney Nano Institute, Sydney, Australia
3 International Center for Materials Nanoarchitectonics (WPI-MANA), National Institute for Materials Science (NIMS), Tsukuba, Japan
4 Graduate School of Pure and Applied Sciences, University of Tsukuba, Tsukuba, Japan
5 The University of Sydney, School of Medical Sciences, Sydney, Australia
6 Brain and Mind Centre, The University of Sydney, Sydney, Australia

* Author to whom any correspondence should be addressed.
E-mail: zdenka.kuncic@sydney.edu.au

Keywords: modularity, multitasking, neuromemristive, nanowire networks, reservoir computing, neuromorphic networks

Abstract

The human brain seemingly effortlessly performs multiple concurrent and elaborate tasks in response to complex, dynamic sensory input from our environment. This capability has been attributed to the highly modular structure of the brain, enabling specific task assignment among different regions and limiting interference between them. Here, we compare the structure and functional capabilities of different bio-physically inspired and biological networks. We then focus on the influence of topological properties on the functional performance of highly modular, bio-physically inspired neuro-memristive nanowire networks (NWNs). We perform two benchmark reservoir computing tasks (memory capacity and nonlinear transformation) on simulated networks and show that while random networks outperform NWNs on independent tasks, NWNs with highly segregated modules achieve the best performance on simultaneous tasks. Conversely, networks that share too many resources, such as networks with random structure, perform poorly in multitasking. Overall, our results show that structural properties such as modularity play a critical role in trafficking information flow, preventing information from spreading indiscriminately throughout NWNs.

1. Introduction

In biological neural networks, higher order functions (e.g. cognition, sensory perception) are thought to emerge from the complexity of the network, and the interplay between structure–function [1–8]. Even the simplest of biological systems such as the nematode C. elegans are capable of processing relatively large amounts of dynamic, diverse, incomplete and even noisy data in real-time in order to navigate their environment [9].

A distinctive feature of biological neural networks is their ability to generalise and perform more than one task simultaneously [10, 11]. For the human brain, such multitasking is generally effortless when we are required to perform low-level tasks, usually involving interaction with our environment (e.g. talking while walking). However, this ability can break down when we attempt to perform two complex tasks, such as making a to-do list while trying to solve an equation [10]. One theory attempting to explain this breakdown in multitasking suggests that if two tasks share similar computational resources (e.g. neural regions), then performing them concurrently leads to interference between the tasks (or crosstalk) and consequently reduced performance [10–14].

Artificial neural networks (ANNs) have been used in a multitasking framework, with the goal to share computational sub-features or processes to improve learning, often referred to as multitask learning (MTL)
[15]. Most commonly, MTL has two major implementations: interactive parallelism (i.e. learning and processing complex patterns simultaneously by considering a large number of interacting constraints), typically implemented in deep learning [16, 17], and independent parallelism (i.e. the capacity to carry out multiple processes independently), typically implemented via parallel distributed computing [11]. These implementations have a fundamental trade-off that has been explored in varying ANN structure architectures, in which the authors alter the number of shared resources to explore generalisability and MTL capacity [10]. MTL in ANNs is typically implemented via sharing of hidden layers, while keeping output layers separate for different tasks [18]. Such implementations in more complex ANNs, including deep neural networks, typically require large amounts of training data, memory, power and specific hardware [19], and struggle to handle noisy multisensory or real-time streaming data [20, 21].

Neuromorphic devices have been developed for implementing ANNs in hardware to improve overall computing efficiency [20, 22–25]. A unique type of neuromorphic system, nanowire networks (NWNs), are particularly interesting as their self-assembly naturally embeds neural network-like circuitry into their structure, with neural-like dynamics emerging from recurrent feedback loops and memristive cross-point junctions [26]. This means neuromorphic NWNs do not require ANN implementation, and instead can be implemented in a reservoir computing (RC) framework. RC bypasses the limitations imposed on ANNs by exploiting the inherent temporal processing capabilities of recurrent neural networks such as NWNs [27]. RC utilises the non-linearity of a dynamical network (reservoir) to map input signals into a higher dimensional feature space such that training is effectively linearised, thus drastically reducing the computational overhead compared to conventional ANN approaches [28]. NWNs are also fault-tolerant to perturbations such as junction failure [29], making them robust reservoir systems.

In our previous study [30], we showed that NWNs exhibit a small-world architecture, similar to the simple biological neural network of C. elegans, albeit more segregated and modular than the brain of the worm. High modularity has also been shown in quasi-3D stacked NWNs [31], although small-worldness appears to be reduced in these representations. Modularity is a characteristic of complex networks and has been shown to be of critical importance for diverse behaviours. For instance, modular networks give rise to more complex dynamics than random networks [32] and promote functional specialisation across complex networks [33]. The organisation of networks into modules also allows for activity in one module with minimal perturbation of any other modules or sections of the network [33, 34]. Consequently, multitasking with minimal resource overlap is plausible in highly modular networks.

Here, we test the functional capabilities of NWNs by simulating networks as reservoirs in an RC set-up, following previous work in both hardware [21, 35] and simulation [36–39]. We focus on the influence of modularity on the performance of NWNs on two RC benchmark learning tasks, non-linear transformation (NLT) [36] and memory capacity (MC) [40]. In particular, we test the hypothesis that due to the similarity of neuromorphic NWN structure with biological neural networks [30], the functional advantage of their highly modular structure may be the ability to perform multiple tasks simultaneously in different modules.

2. Results

2.1. Network comparison

All results are obtained from simulations based on physical and biologically-inspired networks, each modelled with the same memristive synapses. We first compared a self-assembled NWN with three other physically-motivated networks with different topologies: sparse crossbar array, a randomly-rewired NWN (hereafter referred to as a random network) and the C. elegans structural connectome. A crossbar array, typically used in memristor devices [41, 42], has a bipartite network structure. The random network represents an NWN artificially reconstructed to remove any complex network structure, while C. elegans is a biological neural network. The average degree ($\langle k \rangle \approx 13$), number of nodes (between 277–300) and number of edges (between 1800–2100) were kept as similar as possible to the C. elegans network, to best compare the effect of each network’s structure on its function. Graphical representations of the networks are shown in figure 1(A).

Figure 1(B) shows four graph theory metrics used to measure structural properties of each network: network diameter (which is also the shortest path length from source to drain electrode), modularity ($Q$), average clustering coefficient (global clustering) and small worldness (SWP). The NWN (red) exhibits the highest modularity, longest diameter, highest average clustering coefficient and highest small worldness. To investigate the structure–function relationship of these networks, all networks were simulated with the same voltage-controlled memristive edge–junction model, based on conductive filament formation and annihilation (see methods). Figure 1(C) compares the performance of each network on two different benchmark tasks: NLT and MC. Clearly, the NWN (red) performs relatively poorly on both tasks at low voltages ($<1.75$ V), but comparatively well, or even better at $V \gtrsim 1.75$ V. When the NWN is randomly rewired (green), it has significantly lower...
Figure 1. Network comparison. (A) From left to right, graphical representations of a sparse crossbar array network, a random network, a C. elegans network, and a self-assembled NWN. (B) Structural connectivity measures of each network, including network diameter, modularity ($Q$), average clustering and small world propensity (SWP). (C) Comparison of NLT accuracy and MC score for different networks. All networks have similar average degree ($\langle k \rangle \approx 13$), number of nodes ($\approx 300$) and number of edges ($\approx 2000$), as well as the same memristive edge–junction model. Error-bars represent standard error of the mean over ten independent statistical representations of each network (except C. elegans).

modularity, diameter, average clustering coefficient and small worldness, yet its performance on the NLT and MC tasks is generally superior across all voltages, except at $V \lesssim 1.75$ V for the NLT task.

The C. elegans network (yellow) falls between the NWN and its randomly-rewired counterpart in diameter, clustering, modularity and small worldness. It performs comparably to the random network on NLT and worse on the MC task. Compared to the NWN, C. elegans performs better at the NLT task at lower voltages, and comparably at higher voltages, as well as slightly better at lower voltage and significantly worse at higher voltages on the MC task.

Lastly, the sparse crossbar array network (blue) has comparable structural parameters to the random network, with similar diameter, modularity and small worldness, however it shows no clustering at all. The network performs comparably to the random network on the NLT task up to 3 V, above which its performance drops. For the MC task, the sparse crossbar array exhibits a similar trend to the other networks, with performance generally increasing with $V$. It performs better than C. elegans and the NWN over the range 1–3 V and worse than the random network above 2 V.

Overall, the sample self-assembled NWN does not perform as well as its randomly-rewired counterpart on both NLT and MC tasks at lower voltages. Since the random network shares the same degree distribution, number of nodes and number of edges as the NWN, differences in performance must be due to structural parameters, or placement of the electrodes (we show that the latter is not strongly responsible for performance differences in supplementary figure 1 (https://stacks.iop.org/NCE/1/014003/mmedia)).

Figure 2 qualitatively shows functional differences in how the NWN and the random network perform the NLT task. Functional subgraphs showing junction conductance are presented for each network after 10 s of the input signal for three different voltages (see supplementary movie 1 for sample network activation over 10 s for both NLT and MC tasks). For voltage 1, both networks are in a state of low activation. In this state,
Figure 2. Comparison of NLT task accuracy for NWN and random memristive network. (A) Functional graph representations of random snapshots immediately after training for three key voltages: (1) network not yet activated, and performance poor; (2) network activated and performance improves; (3) significant portion of network is active and performance declines. Colourbar represents instantaneous junction conductance, $G$, thresholded at $10^{-6} \, \text{S}$ for clarity of visualisation. (B) Performance on NLT for NWN (red) and random (green) networks. (C) Functional graph representation of NWN snapshots, at the same network states as A. Here, the network states 1 and 2 occur at higher voltages. No conductance paths are formed between source and drain, however paths are beginning to form from each electrode to its closest neighbouring nodes. NLT accuracy for both networks is comparable. Voltage 2 is chosen such that each network performs best (or close to) on the NLT task. Both networks are clearly activated, with one or more high conductance paths formed between source and drain electrodes, but much of the network still remains inactive. For voltage 3, each network is highly active after 10 s, but NLT performance decreases from the peak.

A single path formation between source and drain electrodes (also known as the ‘winner-takes-all path’ (WTA) [43, 44]) has been previously shown to coincide with a sudden increase in network conductance, and correspondingly, task performance [38, 45, 46]. Due to this sharp increase in conductance, we henceforth refer to the WTA pathway formation as the point of network activation. Here, the voltage required to reach activation is higher for NWNs than for random networks (2 V compared to 0.75 V, respectively). As the diameter and average path lengths of NWNs are higher, more voltage is required for the network to form a pathway between the source and drain electrodes. In contrast, since the random network has a smaller diameter, the distance between source and drain electrodes is shorter, so less voltage is required to activate a conducting path between source and drain.

When we control for path length between source and drain in the NWN, while network activation still occurs at a lower voltage (around 1.25 V) compared to the random network, a noticeable difference in network performance persists; i.e. the NWN performs significantly worse (see supplementary figure 1). The decrease in performance can be attributed to a considerably smaller portion of the NWN being activated, relative to the random network, in which much of the network is activated, even though the path length between source and drain.
Figure 3. Heatmap of mean NLT accuracy (top) and mean MC score (bottom) for NWNs of varying density (columns), across different input voltages (rows). Density is represented by the average degree of each network, and increases from left to right. The horizontal axis labels at the bottom of the MC plot represent the average degree of each corresponding column. The networks shown in between the two heatmaps are example graphical representations of increasing average degree. Each pixel represents NLT or MC performance averaged across ten unique networks for each density/voltage pairing.

Drain is the same in both networks. This stark difference in performance can be largely attributed to differences in network structural connectivity.

As random networks have relatively low modularity, negligible clustering, and low small worldness (cf figure 1), the possible paths to traverse between source and drain are less restricted and therefore more numerous than in NWNs. As such, conductance can readily spread to the closest connected nodes as voltage increases. This allows for much of the network to be activated in significantly fewer steps. In contrast, NWNs are structurally constrained by a more modular structure, with higher clustering and longer path lengths between different sections (communities) of the network. This structure restricts information flow to the closest neighbours of the activated nodes, requiring more steps (and higher voltage) to reach distant parts of the network.

To further explore the effect of structural parameters (including modularity) on NWNs and their performance on benchmark tasks, we performed a parameter sweep across different network topologies. In the following sections, we explore the effect of varying network density and modularity on NLT and MC task performance.
Figure 4. Sample NWN activation and task performance at varying densities. The input voltage for all networks shown is 1.75 V. A ‘WT A’ path is evident in the top row, second column (avg. deg 13.5, NLT task), and bottom row third column (avg. deg 27, MC task). At densities below this, no path is formed between source and drain, and at densities above, many more pathways are activated. All snapshots are taken at a readout time of $t = 10$ s. Colorbar shows junction conductance.

2.2. Density
In the network comparison above, network density was kept constant (with $\langle k \rangle \approx 13$) across the four different types of networks. Here, we explore the change in performance on NLT and MC tasks when the average degree of NWNs is varied. To do so, we constructed networks of $\sim 300$ nodes, with an average degree ranging from 5 (highly sparse) to 286 (highly dense). For more information about network construction, see methods. Importantly, at very high densities we see a vast reduction in small worldness (cf table 2). This was also shown in a quasi-3D model of NWNs [31].

Figure 3 shows a heatmap of average performance across ten networks for each network density (columns) increasing from left to right, with varying input voltage (rows). The top panel shows mean accuracy on the NLT task, and the bottom panel shows mean MC score.

At low densities, high voltage is required to drive the network into activation (based on Kirchoff’s circuit laws). This is due to longer path lengths between source and drain, and fewer parallel paths for the voltage to transverse [46, 47]. As density increases, so too does the number of connections and parallel paths, so that lower voltage is required to activate the network. For the NLT task, this results in higher performance at higher densities and lower voltages. Nonetheless, above a certain density, performance decreases. This is due to too much of the network being activated, similar to the effect observed in the functional subgraphs in figure 2 at high voltage. Figure 4 shows the corresponding functional subgraphs for varying density. At high density, the signal spreads indiscriminately throughout the network and limits NLT accuracy (and MC score, but at higher density—cf figure 3). At low to medium densities, when the network is activated, only sections of the network that are on or near the WTA pathway are activated. Both NLT and MC performance improves dramatically when the WTA is formed. We have shown this to be associated with a first order phase transition [47]. While some parallel or forking pathways may be seen, much of the network remains inactive. The competition for pathway formation ensures a diverse variety of junction dynamics and hence, richer and more separable output features that serve to improve regression to the target waveform in the NLT task [27, 48].

For the MC task, figure 3 shows a more consistent improvement in performance as density increases, up to a level (around 256–271) above which catastrophic failure occurs. Below this, performance peaks at an average degree of around 97 for a range of voltages (cf figure 4). Why are these results so different from the NLT task? The MC task is inherently different from the NLT task: output features are regressed to a target signal (the input signal delayed by varying amounts) that is random and more rapidly varying than the periodic target signal used in NLT. Thus, both tasks involve different timescales and use different reservoir resources. MC relies mostly on the reservoir’s fading memory due to the recurrent connections, whereas NLT relies mostly on the resistive memory of junctions, which has slower dynamics [27]. Thus, as density increases with the number of junctions, it becomes increasingly more difficult for the reservoir to remember high frequency components of the input signal as the reservoir resources become overwhelmingly dominated by the lower frequency dynamics of resistive memory.
2.3. Modularity and multitasking

To test the hypothesis that modularity enables multitasking, we segmented the original NWN into two separate modules (see methods for the full network construction process). We then randomly rewired edges from the two separate modules in six steps, decreasing modularity from two, fully-separated modules to a network with two highly segregated modules, to a network with two highly integrated modules (similar to the random network in section 1). We repeated this process for ten total NWN realisations. The panels beneath figures 5(A) and (B) show the different modularity values, from fully integrated (left) to fully segregated and separated (right), where the modules are not connected at all.

We performed the NLT and MC tasks simultaneously in each of the two network modules. Figure 5(A) plots the mean NLT accuracy and mean MC score as a function of input voltage for varying levels of modularity. Figure 5(B) plots the corresponding normalised multitasking score for both tasks (see methods for details on how this score is calculated). Figure 5(C) compares the normalised multitasking score for the three cases of a fully integrated network, a highly segregated network and separated network modules (respectively corresponding to the first and last two columns of figure 5(B)).

From figure 5(A), the NLT task performs best when the network modules are fully separated. Although the highly segregated network performs comparably well, performance drops abruptly from fully separated to highly integrated, with the most integrated networks generally performing significantly worse at NLT. Contrastingly, for the MC task, performance decreases catastrophically for fully separated modules compared to highly segregated networks. However, integrated networks slightly outperform segregated networks, particularly at higher voltages.

When considering multitasking performance (figure 5(B)), segregated networks outperform integrated networks at almost all voltages (except for 0.5 V). A sharp drop in performance is evident in fully separated network modules compared to highly segregated networks. Indeed, as figure 5(C) shows, fully separated network modules, with no connections, perform considerably worse at multitasking than either highly segregated or integrated networks. The overall best performance on multitasking is found for segregated networks.

To test if the modular structure is directly responsible for this result, we swapped the source electrode from module 2 with the drain electrode of module 1, such that both sources are in one module, and both drains in the other. Under this configuration, performance drops drastically for the segregated networks, (see supplementary figure 2). This confirms that for modular structure to be useful for multitasking, each task must
be allocated to a dedicated module. Moreover, our results suggest that some level of crosstalk between modules is necessary to achieve optimal multitasking performance.

To better understand why segregated networks perform better than integrated networks at multitasking, we investigated NWN activation and functional sub-graphs for different voltages at $t = 10$ s (see supplementary materials for the full time-series video). Figure 6 compares multitasking performance for segregated (green) and integrated (red) networks, highlighting three key voltages to visualise network activation. These three voltages reflect different states of the networks, the same as in figure 2. The integrated network performs best at a lower voltage, $V = 0.5$ V, which can be attributed to paths being able to form more readily than in the segregated network. At this voltage level, there is negligible interference between the two tasks, as not much of the integrated network is active, leaving much of the network’s resources untouched. At higher voltages, when much of the network is activated, a larger amount of overlap between the two tasks is evident, causing interference, crosstalk and more competition for the same resources. In contrast, the segregated network requires higher voltage for enough of the network to be activated to perform multitasking well. Since the two modules are highly segregated, even at 5 V there is insufficient interference between the tasks to impact performance significantly. Furthermore, different resources are largely being employed for each task, with minimal overlap. A reduction in multitasking performance is only evident at 10 V, at which point most of network is activated, and many more of the same resources are being shared.

Figure 6. Multitasking functionality and performance comparison in segregated and integrated NWNs. (A) Functional sub-graph representations of a sample of segregated networks, activated by three key voltages: (1) network not yet activated, and task performance poor; (2) NII module (left module) of network is activated and the network performs well on both tasks (3) paths are formed in both modules from source to drain and network performs best on both tasks. (B) Multitasking performance for segregated (green) and integrated (red) networks, at varying voltages. (C) Functional sub-graph representations of a sample of integrated networks, activated by the same voltages as A: (1) network exhibits a number of active pathways and performance is higher than segregated network; (2) a significant amount of the network is activated, but performance decreases; (3) a large amount of network is activated, but performance does not improve significantly. Colorbar represents junction conductance, $G$. Conductance is thresholded at $10^{-6}$ S for clarity of visualisation.
What does interference between the two tasks in such cases look like? When both tasks share too many of the same resources, they are not equally affected by the resulting interference. Performance on the NLT task drastically decreases, while MC remains relatively unaffected. These changes in performance may be attributed to the difference in timescales between the two input signals, as the NLT task relies on a slower timescale while the MC task fluctuates more quickly. A slower timescale (NLT) enables conductive junction filaments to form/decay more consistently, whereas rapid random fluctuations (MC) does not allow for this. As such, even with only a few connections between modules, the MC task acts as a source of increased noise to the junctions in the NLT module, creating instability in filament formation and decay. This is borne out when controlling the voltage of the NLT task while increasing the MC voltage (cf supplementary figure 3). As MC voltage increases, NLT accuracy decreases sharply even if the NLT input voltage is kept constant. Contrastingly, for the MC task, the NLT input acts as a slower modulating signal, increasing MC performance. Analysis of constant voltages further supports the notion that the MC task hinders performance on NLT at higher voltages due to increased crosstalk, and the NLT task boosts performance on the MC task at higher voltages.

Finally, based on the density results showing both NLT and MC utilising different timescales, we wanted to ensure that multitasking results were not improved simply because each task had access to more network resources (i.e. 300 nodes instead of 150 within their own module). Supplementary figure 4 shows that access to more resources is not enough to improve MC to the same levels we see in multitasking, and that NLT acts as a modulating signal to increase MC performance.

Overall, these results suggest that multitasking is most effective in highly modular networks that allow some level of crosstalk between the segregated modules. This, in turn, enables some optimal resource sharing yet limits competition for resources between the modules.

3. Discussion

Our results show that limiting information processing of a specific task to its own module is highly advantageous, improving performance on multitasking as long as few connections are available between modules. This is consistent with previous findings which demonstrate that it is more advantageous for biological networks to perform non-linear computations within a module, and then pass on the information to the rest of the system [49, 50]. Other RC implementations based on neural brain circuits have shown similar improved performance due to a modular organisation, linking such topology with critical dynamics [51]. Performance advantages in modular networks have also been shown in ANNs [52], and neural networks have demonstrated emergence of segregated communities in pattern recognition tasks, similar to the visual cortex [53].

Biological networks rely on a hierarchical, modular architecture to utilise minimal resources [49, 50, 54]. Similar to other biological neural networks, the human brain is likely comprised of a series of globally sparse, hierarchical modular networks [55]. Segregated, hierarchical networks are thought to be important for extracting sensory information from the environment [54]. Structured, long-range connections between modules help improve computational performance and efficiency, reduce response variability, increase robustness against interference effects, and boost MC [50]. Projections between modules lead to improvements by allowing information to propagate to deeper modules within a hierarchical structure [50]. Such long-range projections between segregated brain regions have been shown to allow transmission of local information to distant cortical regions in mice [56]. While the NWNs studied here do not have a hierarchical structure, our results confirm functional advantages of modularity are similar to the brain. Further investigations are warranted to determine whether additional functional advantages may be realised in NWNs with hierarchical as well as modular structure.

3.1. Network resource allocation: NLT and MC task performance

For the two RC benchmark tasks considered here, NLT and MC, we found that NWNs generally require higher voltage or density (i.e. average degree) to perform comparably to random networks, due to their higher modularity, average clustering and small worldness. Our results indicate that these structural properties play an important role in trafficking network information flow, preventing information from spreading indiscriminately throughout the network.

In previous studies, the NLT waveform regression task was tested on atomic switch NWNs both experimentally [21] and in simulation [36]. To perform the NLT task, Demis and colleagues [21] input a sine wave into a NWN hardware device and linearly regressed multiple electrode readouts to a square wave with accuracy $\approx 73\%$. They reported a network density of $\approx 10^9$ junctions cm$^{-2}$, and a network size of $\approx 4$ cm$^2$. It is difficult to compare simulation results from the current study using 300 nanowires to those obtained with the densities of Demis et al, however the simulation study by Sillin et al [36] provide a better comparison. They did not report accuracy, but reported the lowest mean square error (MSE) of $\approx 0.12$, using a similar density
Table 1. MC performance in literature via different implementations of ESNs, compared to NWN performance.

| Study                                      | Network size | Avg. degree | Max MC | Implementation                  | Neuron type   |
|--------------------------------------------|--------------|-------------|--------|---------------------------------|---------------|
| Barančok and Farkaš (2014) [57]           | 150          | N/A         | 90     | Edge-of-chaos ESN               | Sigmoidal     |
| Farkaš, Bosák and Gergeľ (2016) [58]      | 255          | Low         | 64     | Sparse ESN                      | Sigmoidal     |
| Farkaš, Bosák and Gergeľ (2016) [58]      | 100          | Low         | 45     | Sparse ESN                      | Sigmoidal     |
| Kawai, Park and Asada (2019) [59]         | 500          | 6           | 35     | Small-world ESN                 | Sigmoidal     |
| Rodriguez, Izquierdo and Ahn (2019) [60]  | 500          | 6           | 10     | Modular ESN                     | Sigmoidal     |
| Here                                      | 150          | 15          | 26     | Modular NWNs                    | Memristive    |
| Here                                      | 300 (readout from 150) | 15 | 93     | Modular + multitasking NWNs    | Memristive    |
| Here                                      | 300          | 97          | 120    | Optimal density NWNs            | Memristive    |
| Here                                      | 300          | 6           | 60     | Comparable density NWNs         | Memristive    |

(to Demis et al.), however they only simulated 50–400 total connections in the network. Results from the current study show higher square-wave NL T performance of up to 85% (see figure 3) with a significantly reduced number of wires. The improved performance in square-wave NL T likely arises from the topology of our simulated NWNs. Both Demis et al and Sillin et al grew nanowires around grid-shaped copper posts, while our simulation models self-assembled networks. It is possible that the grid-like structure of the copper posts also affects the topology of NWNs.

The MC task was originally proposed specifically for echo state networks (ESNs) [40] and has been tested on networks with different structures [57–60]. Table 1 lists the maximum MC scores attained with different ESNs, compared to MC scores for NWNs with similar properties. The best performance, with a maximum MC of 120, is achieved for an NWN with 300 nodes and average degree of 97. When the average degree is reduced to 6, maximum MC drops to 60, which is still substantially higher than that achieved for a 500-node ESN with the same density. The best performing ESN in this list is for a network prepared in an edge-of-chaos state. While we did not pre-initialise NWNs in this study, we have elsewhere shown that NWNs can be prepared in an edge-of-chaos or critical-like state to improve performance in RC tasks [47, 48]. In those studies, we found a maximum MC of around 12 for 100 nanowires and 100 for 300 nanowires [48], and NL T accuracy ≈ 0.85 with 100 nanowires [47].

Modular NWNs with 150 nodes in each module achieve poor MC performance when modules are fully separated, however once even a small amount of overlap is allowed (i.e. segregated networks), the NWNs outperform even the best ESN (with max MC of around 90). When compared to modular ESNs [60], this is a large improvement in performance. This difference in performance may be attributed to the internal memory property of memristive junctions, which allows NWNs to outperform ESNs, which typically use mathematical sigmoid functions devoid of memory [61]. It is important to note that some ESNs implement integrator neurons, which provide memory at the neuron level. However, to our knowledge, such implementations in the literature have yet to test either MC or NL T tasks.

An advantage of modularity in complex networks is allocation of network resources [49, 50], which allows segregated networks to vastly outperform random (integrated) networks in multitasking [11]. Such effortless multitasking becomes troublesome when network resources are shared (e.g. when performing two complex logical tasks) [10]. If too many network resources are shared, as in the case of highly integrated networks, multitasking becomes less efficient due to interference or crosstalk [10, 11]. We observe this modular advantage here, with segregated NWNs outperforming random memristive (integrated) networks when multitasking, but not on independent tasks.

The NL T task performs NL T of the input signal by the network. This transformation is achieved via a nonlinear activation function at the memristive junctions once a conductive filament bridge forms [47]. Our results show that when enough junctions are activated to form a signal transduction path from source to drain, the network performs optimally on the task. This state produces an optimal number of linearly separable outputs in the feature space. In a physical NWN, this corresponds to significant inhomogeneities in the voltage distribution, with just enough junctions activated to nonlinearly transform the signal. This is qualitatively consistent with the network being in a critical-like dynamical state [38, 47]. Conversely, at higher densities or voltages, too many junctions are activated and the outputs are not sufficiently linearly separable (i.e. reduced feature space). In the physical network, this corresponds to a homogeneous voltage distribution. This results in failure of the network to perform NL T, which we observe from avg. degree of 170, or at voltages higher than 5 V (for avg. degrees greater than 20).
In contrast to the NLT task, the MC task does not rely on the internal resistive memory of individual junctions, but instead relies on the fading memory of the recurrent network connections [40]. Thus, each task tests the network’s ability to recall information from the input signal on different timescales, with reservoir-level fading memory depending on faster timescales than junction-level resistive memory. As such, we see higher MC performance at much higher densities and voltages, which supports findings in our previous work [48]. This also explains NWNs’ ability to outperform typical ESN implementations with sigmoidal neurons that lack memristive memory properties [27].

3.2. Multitasking
Our results show that some level of crosstalk is beneficial for multitasking in NWNs. This supports the results of other studies on neural network architectures showing effortless multitasking when network resources are not overly shared, and reduced multitasking capacity when too many similar resources are being shared [10, 11]. The MC and NLT tasks applied here operate at different time scales, and rely on different network resources. When applied simultaneously to a modular network, the NLT acts as a modulating signal for the MC, boosting performance. This increased performance is similar to findings from implementations of MTL in RC. Such implementations involve parallel arrays of reservoir networks [62], which show improved performance on the MC task over single reservoir network implementations [63, 64]. However, an important feature of our NWN is the additional resistive memory of individual edge–junctions, which enables the network to perform temporal signal processing tasks involving vastly different timescales.

Our results support the idea introduced by Musslick et al [10] and Petri et al [11] that interference reduces multitasking performance when too many resources are shared (i.e. overlapping modules). We show that interference need not be detrimental and may in fact be advantageous if resources are only marginally shared between modules via sparse adaptive connections (i.e. segregated modules).

3.3. Conclusion
Results presented here extend our previous findings [30] that NWNs are more modular than random networks, C. elegans and crossbar arrays, by showing that network structure impacts function. In particular, our results demonstrate the advantage of these modular structures for multitasking, as suggested in brain network studies [49, 50, 54, 55]. These results motivate future implementation of multiple tasks in highly modular networks, in an effort to implement bio-inspired multitasking in other neuromorphic systems. Implementing modular structures in hardware NWNs may be advantageous for performing multiple simultaneous tasks while maintaining low power requirements.

4. Methods

4.1. Network construction

4.1.1. Network comparison
We compared four networks (C. elegans, sparse crossbar, NWNs and random networks), each representing a different type of real-world network. For each network type except for C. elegans we constructed ten unique networks, each sharing the same parameters but with different random seeds.

As in our previous study [30], the C. elegans structural connectivity matrix (277 neurons and 2105 synaptic connections) was adapted from Achacoso and Yamamoto [65], and electron microscope reconstructions by White and colleagues [66].

The sparse crossbar arrays were constructed using the bipartite module from the NetworkX algorithms package [67], keeping the number of nodes and edges as close as possible to the C. elegans structure. This array represents a sparse implementation of cross-bar architectures.

The sample NWN networks were constructed as described in our previous study [30] and was selected from a range of varying networks as they had the most similar number of nodes and edges as the C. elegans.

Edges were modelled as threshold-driven bipolar memristive switches, as described briefly in equation (1) and in more detail elsewhere [37–39, 47, 68, 69]. At each junction, we model electrochemical metallisation via a conductive filament parameter \( \Lambda(t) \). The filament parameter is restricted to the interval \(-\Lambda_{\text{max}} \leq \Lambda(t) \leq \Lambda_{\text{max}}\) and dynamically evolves according to equation (1):

\[
\frac{d \Lambda}{dt} = \begin{cases} 
(V_{\text{reset}}(t) - V_{\text{reset}}) \left( \text{sign}(V_{\text{reset}}(t)) \right) & V_{\text{reset}}(t) > V_{\text{reset}} \\
0 & V_{\text{reset}} < |V(t)| < V_{\text{reset}} \\
|V_{\text{reset}}(t)| & V_{\text{reset}} > |V_{\text{reset}}(t)| \\
0 & |\Lambda| \geq \Lambda_{\text{max}}
\end{cases}
\]
method, as more edges were chosen than there were junctions in the network, some edges were swapped more than once. This maintained a constant average degree and degree distribution across the varying modularities.

Structural properties were assigned the same ten random seeds, one for each network. This allowed for variation within modularity groups (besides their modularity). Structural properties for each modularity group are shown in Table 3.

Parameters were chosen such that network activation time was comparable to hardware networks. Values used are $V_{in} = 0.01$ V, $V_{reset} = 0.001$, and $\Lambda_{max} = 0.15$ Vs. $b = 10$ is a constant defining the decay of the filament. $V_{in}$ represents voltage across each junction at time, $t$.

All junctions are initially in a high resistance ‘$R_{off}$‘ state ($\Lambda = 0$ Vs). For each junction in the network, resistance switches to ‘$R_{on}$‘ state when $\Lambda \geq \Lambda_{crit}$ where $\Lambda_{crit} = 0.10$ Vs is a set threshold. The ratio of these resistance states is $R_{off}/R_{on} = 10^3$, with $R_{on} = G_0^{-1}$, and $G_0 = (13 \text{k}\Omega)^{-1}$ is the conductance quanta [48]. Junction electron tunnelling conductance was also modelled as explained in Hochstetter et al [47].

Random networks were constructed using double-edge-swaps using the NetworkX package. Two pairs of two nodes, each with an edge between them, were randomly selected and the edges were randomly swapped between the nodes. This process was performed 50,000 times to ensure that almost all the nodes and edges were rewired. Any self-loops resulting from this process were removed. This process also maintains the degree distribution of the original NWN networks. Consequently the random networks had the same number of nodes, edges and degree distribution as the NWN networks, but with a vastly changed structure (approximately random). For comparison, the average degree of each of the networks was constrained to around 13, the density of the biological C. elegans network.

### 4.1.2. Density

Networks with varying density were constructed from 200 NWNs, each with 277–300 nodes, to ensure similarity to the original sampled NWN network above. To create a range of densities, network parameters were kept constant (e.g. average wire-length $= 10 \mu$m), while decreasing the size of the simulated 2D space in which the networks were placed. As such, networks placed within a large 2D space have lower density, while networks placed within a small 2D space have higher density. Through this method, 20 different densities were created, each of which had ten networks with unique random seeds (e.g. ten networks with average degree of five had unique seeds [1, 2, 3, …, 9, 10], and ten networks with average degree of ten had the same seeds [1, 2, 3, …, 9, 10]).

Structural properties for each density group are shown in Table 2.

### 4.1.3. Modularity

To construct networks with varying modularity, 60 NWNs were created. For each NWN, we created two network realisations comprised of around 150 nodes each. Each realisation was used as a module in the network. This resulted in two modules and 277–300 total nodes for each network.

When referring to separated networks, we refer to each of the two modules in each network, with zero overlap or connections between each of them. For segregated networks, we introduced a small number of overlapping connections between each module. In order to capture a range of modularities (i.e. from fully segregated to fully integrated), we performed random edge rewiring of the networks. First we selected a number of edges, $e$, for rewiring. We increased this number exponentially, from segregated to integrated in six groups, $e = [1, 4, 24, 121, 601, 2980]$. We randomly swapped the nodes which these edges were attached to. For this method, as more edges were chosen than there were junctions in the network, some edges were swapped more than once. This maintained a constant average degree and degree distribution across the varying modularities.

For each of the six modularity groups, ten networks were created with different random seeds. All groups were assigned the same ten random seeds, one for each network. This allowed for variation within modularity groups, but minimal variation between modularity groups (besides their modularity). Structural properties for each modularity group are shown in Table 3.

### Table 2. Avg. degree, small worldness and modularity of varying density networks.

| Avg. deg | 5.16 | 5.84 | 6.56 | 7.54 | 8.51 | 9.67 | 11.6 | 13.5 | 16.9 | 20.4 | 27.0 | 34.2 | 44.7 | 67.7 | 97.6 | 170 | 238 | 256 | 271 | 285 |
|----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-----|-----|-----|-----|-----|
| Small worldness | 0.74 | 0.72 | 0.71 | 0.71 | 0.70 | 0.69 | 0.69 | 0.68 | 0.67 | 0.66 | 0.65 | 0.65 | 0.64 | 0.61 | 0.48 | 0.28 | 0.28 | 0.27 | 0.26 |
| Modularity | 0.82 | 0.79 | 0.78 | 0.75 | 0.73 | 0.71 | 0.68 | 0.65 | 0.62 | 0.59 | 0.53 | 0.49 | 0.44 | 0.35 | 0.25 | 0.10 | 0.02 | 0.01 | 0.00 | 0.00 |

### Table 3. Avg. degree, small worldness and modularity of varying modularity networks. The leftmost column refers to integrated networks, and the rightmost refers to segregated networks.

| Modularity (Q) | 0.63 | 0.63 | 0.62 | 0.54 | 0.32 | 0.20 |
|---------------|------|------|------|------|------|------|
| Avg. deg | 15.9 | 15.9 | 15.9 | 15.9 | 15.9 | 15.9 |
| Small worldness | 0.67 | 0.68 | 0.67 | 0.57 | 0.35 | 0.31 |
4.1.4. Multitasking
The modules described above were used to implement multitasking, as follows. Source–drain electrode pairs were placed in each module, such that a different task could be implemented into each module simultaneously. To calculate the nodes belonging to each module, we used the community Louvain algorithm from the brain connectivity toolbox (BCT) [70], with a gamma of 0.1 to ensure two large modules were identified. We selected all nodes within an individual module as read-out nodes for each of the NLT and MC tasks, ignoring all nodes in the other module. For example, in a 300-node network, if module 1 had 150 nodes, we input the NLT task into module 1, and read out from nodes 1–150. We would simultaneously feed in the MC task to module 2, which also had 150 nodes. For the MC task, we read out from only nodes 151–300, ignoring the nodes in module 1.

While it is relatively straightforward to separate two modules in a highly segregated network, this becomes more difficult as a network is rewired to become increasingly more integrated. Therefore, we kept the same node IDs from the segregated network across all networks, and read out from those nodes for each task across all modularities. If module 1 in the segregated network had node IDs 1–150 for the NLT task, we kept reading out only from nodes 1–150 in the more integrated networks too. We also kept the node ID for source–drain electrodes consistent across network modularities.

4.2. Reservoir computing tasks
All simulations were conducted on Python v3.7.3 and Matlab v2020a. The networks were used as reservoirs in a RC setting using two RC benchmark tasks: NLT [36] and MC [40]. These tasks leverage different aspects of a reservoir’s memory properties: NLT uses a relatively slowly varying continuous input signal that preferentially selects for the internal resistive memory of individual junctions, whereas MC uses noisy, randomly selected inputs that preferentially select for the recurrent network’s fading memory [27]. We previously implemented these tasks with NWNs in different contexts [37–39, 48]. The RC tasks were implemented by defining one input (source) node, one grounded (drain) node, and all other network nodes as output nodes. Using the voltages at the output nodes as a readout, we trained a linear regression model using only the readout to fit the respective target signal.

For multitask training using modular networks, we treated each module as a separate network, with one source–drain pair in each module and the rest of the module nodes used as output nodes. This meant that a modular 300-NWN would have two sources, two drains, and around 150 nodes in each module used as readouts for separate tasks (e.g. module 1–150 nodes for MC task, module 2–150 nodes for NLT task)

4.2.1. Nonlinear Transformation
NLT is a waveform regression task that non-linearly transforms a continuous, slowly varying sinusoidal input signal and linearly regresses the outputs to a different waveform [21, 36]. Using an input sine wave with frequency $f = 0.5$ Hz and varying amplitudes, $V = 0.2, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2, 3, 5, 10 \text{V}$, in a voltage sweep. Linear combinations of the node outputs were used to regress to a square wave target signal. Accuracy is calculated as 1-RNMSE, where RNMSE is root-normalised MSE For further details, see Fu et al (2020) [37] and Zhu et al (2021) [48].

4.2.2. Memory capacity
MC evaluates a reservoir’s ability to recall past information by reproducing delayed versions of a uniform random noise input signal [40]. Input voltage signals were generated from uniform random samples in the interval $[-V, V]$, where $V$ is the voltage amplitude chosen from varying amplitudes, $V = 0.2, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2, 3, 5, 10 \text{V}$. Linear combinations of the network readouts at time $t$ were used to regress to the input signal at the earlier time $t - n$, where $n$ ranges from 1 to the size of the network (277–300). For further details on how we implemented this task, see Zhu et al (2021) [48].

4.3. Graph theory measures
To measure the structural connectivity of each network, the BCT [70] and NetworkX [67] packages were used. To calculate modularity, we used the community Louvain package of the BCT, based on the Louvain community detection method [71]. The optimal community structure of NWNs was determined by performing a resolution sweep on three sample NWN networks, from which we found that $\gamma = 1.1$ best captures the modules in our networks. The exception to this was for the two highly segregated modules used for multitasking, for which we used $\gamma = 0.1$.

Acknowledgements
The authors acknowledge the use of the Artemis High Performance Computing resource at the Sydney Informatics Hub, a Core Research Facility of the University of Sydney. This research was supported by an Australian Government Research Training Program (RTP) Scholarship.
Author contributions statement

Data collection, analysis, writing and editing was conducted by AL, with support from RZ and JH. ADA and TN helped with revisions and experimental verification of the simulation model. JMS and ZK supervised the project and provided writing, editing and feedback for the manuscript.

Conflict of interest statement

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

ORCID iDs

Alon Loeffler  https://orcid.org/0000-0002-8866-3073
Joel Hochstetter  https://orcid.org/0000-0003-3460-7305
Adrian Diaz-Alvarez  https://orcid.org/0000-0003-4638-8488
Tomonobu Nakayama  https://orcid.org/0000-0001-9696-475X
James M Shine  https://orcid.org/0000-0001-1762-5499
Zdenka Kuncic  https://orcid.org/0000-0001-6765-3215

References

[1] Angeline P J 1994 Genetic programming and emergent intelligence Advances in Genetic Programming vol 1 K E Kinnear (Cambridge, MA: MIT Press) pp 75–98
[2] Hagmann P et al 2008 Mapping the structural core of human cerebral cortex PLoS Biol. 6 1479–93
[3] Bullmore E and Sporns O 2009 Complex brain networks: graph theoretical analysis of structural and functional systems Nat. Rev. Neurosci. 10 186–98
[4] Chialvo D R 2010 Emergent complex neural dynamics Nat. Phys. 6 744–50
[5] Bassett D S and Gazzaniga M S 2011 Understanding complexity in the human brain Trends Cogn. Sci. 15 200–9
[6] Pascual-García A 2016 A topological approach to the problem of emergence in complex systems (arXiv:1610.02448)
[7] Dumitrescu A T, Oltean E, Merezeanu D and Dobrescu R 2017 Emergence in hierarchical complex systems structured as social networks Proc—2017 21st Int. Conf. Control Systems and Computer, CSCS (Bucharest, Romania: Institute of Electrical and Electronics Engineers Inc.) pp 426–31
[8] Lynn C W and Bassett D S 2019 The physics of brain network structure, function and control Nat. Rev. Phys. 1 318–32
[9] Metaxakis A, Petratou D and Tavernarakis N 2018 Multimodal sensory processing in C. elegans Open Biol. 8 180049
[10] Muslick S et al 2016 Multitasking capability versus learning efficiency in neural network architectures Cognitive Science Society pp 829–34
[11] Petri G et al 2021 Topological limits to the parallel processing capability of network architectures Nat. Phys. 17 646–51
[12] Navon D and Gopher D 1979 On the economy of the human-processing system Psychol. Rev. 86 214–55
[13] Meyer D E and Kieras D E 1997 A computational theory of executive cognitive processes and multiple-task performance. Part 1 Basic mechanisms Psychol. Rev. 104 3–65
[14] Salvucci D D and Taatgen N A 2008 Threaded cognition: an integrated theory of concurrent multitasking Psychol. Rev. 115 101–30
[15] Caruana R 1997 Multitask learning Mach. Learn. 28 41–75
[16] Gropp W, Lusk E, Doss N and Skjellum A 1996 A high-performance, portable implementation of the MPI message passing interface standard Parallel Comput. 22 789–828
[17] Bengio Y, Courville A and Vincent P 2013 Representation learning: a review and new perspectives IEEE Trans. Pattern Anal. Mach. Intell. 35 1798–828
[18] Ruder S 2017 An overview of multi-task learning in deep neural networks (arXiv:1706.05098)
[19] Changpinyou S, Sandler M and Zhmoginov A 2017 The power of sparsity in convolutional neural networks (arXiv:1702.06257)
[20] Akopyan P et al 2015 TrueNorth: design and tool flow of a 65 mW one million neuron programmable neurosynaptic chip IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 34 1537–57
[21] TrueNorth: design and tool flow of a 65 mW one million neuron programmable neurosynaptic chip IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 34 1537–57
[22] Demis E C et al 2016 Nanoarchitectonic atomic switch networks for unconventional computing Japan. J. Appl. Phys. 55 110282
[23] Merolla P A et al 2014 A million spiking-neuron integrated circuit with a scalable communication network and interface Science 345 668
[24] Davies M et al 2018 Loihi: a neuromorphic manycore processor with on-chip learning IEEE Micro 38 82–99
[25] Wunderlich T et al 2019 Demonstrating advantages of neuromorphic computation: a pilot study Front. Neurosci. 13
[26] Zdenka Kuncic and Nakayama T 2021 Neuromorphic nanowire networks: principles, progress and future prospects for neuro-inspired information processing Adv. Phys. X 6 1894234
[27] Verstraeten D, Schrauwen B, D’Haene M and Stroobandt D 2017 An experimental unification of reservoir computing methods Neural Netw. 20 391–403
[28] Tanaka G et al 2019 Recent advances in physical reservoir computing: a review Neural Netw. 115 100–23
