Investigating in-vitro neuron cultures as computational reservoir

Peter Aaser

***Abstract*—Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.**

***Keywords*—neurons, reservoir-computing, cyborg**

I. Introduction

Since the 50s, the Von-Neumann computer architecture has been ubiquitous in the field of computing. This architecture owes its success to its relative simplicity, allowing quick iterations in tact with doubling transistor counts in accord with moores law. However several issues with the Von-Neumann architecture have become more and more pressing in recent times. Issues such as the Von-Neumann memory bottleneck are becoming increasingly difficult to hide, and its inherently serial nature of computation. Even in the case that moores law will continue, we still face issues with lack of scalability in modern processors. With billions of transistors a top down design process is becoming increasingly difficult and expensive, and a single faulty transistor can cause an entire chip to be useless. These weaknesses necessitates going beyond the Von-Neumann architecture, exploring unconventional computing paradigms. Taking inspiration from nature we look at cellular computing consisting of computational cells which by themselves are unremarkable. In these systems we observe properties that cannot be traced back to a single cell, they only appear when multiple cells are interacting with each other. In nature these systems form without any form of supervision, a property known as *self organization*, with an example being DNA which "describes how to build the system, not what the system will look like" [1]. These *emergent properties* arise in systems built from the bottom up by a set of growth rules rather than a designers intent, a property, which allows a flexibility and robustness that modern processors lack, a prime example being the human brain. The human brain is a vastly parallel computer, eclipsing modern processors in terms of computational capacity, robustness, energy efficiency and complexity. The three first properties make them very appealing subjects to study for unconventional computing, however due to the immense complexity the brain is still enigmatic. In this paper we explore using neuron cultures to create an organic mechanical hybrid organism, a so called *cyborg*.

II. Background

A. The NTNU Cyborg Project

The NTNU cyborg project is a collaboration between several departments at NTNU including the department of biotechnology, computer and information science, engineering cybernetics, neuroscience and more. [**?**] The stated goal for the cyborg project is "to enable communication between living nerve tissue and a robot. The social and interactive cyborg will walk around the campus raising awareness for biotechnology and ICT, bringing NTNU in the forefront of research and creating a platform for interdisciplinary collaborations and teaching." Currently the department of neuroscience is growing neuron cultures which are to be used as the biological part of the robot. These neuron cultures are not part of a brain, they are fully dissociated, grown in special chambers in-vitro. The robot part of the cyborg has been developed and implemented by the department of engineering cybernetics, and is currently operational, however it has not yet been integrated with the in-vitro neuron cultures. The challenge faced by the cyborg project is the infrastructure for interfacing the neuron cultures and the robot, essentially creating a brain computer interface.

B. Complex Systems

In this paper references are made to computations done by both artificial and real neurons. To make the distinction between these cases clear all computation done by computer simulated approximations of neurons will be prefixed as artificial.

The self-organizing properties that makes cellular computing a such a successful paradigm in nature also make understanding and exploiting them difficult. Many such systems have been explored, such as recurrent artificial neural networks [2], random boolean networks [3] and cellular automata. In [4]

Langton explores the requirements for systems to support emergent computation, where he argues that in order for a system to support emergent computation it must lie in a *critical phase* between order and chaos, drawing parallels to phase transitions in material science. This observation comes from thermodynamics, in which a material may exhibit a *second order phase transition* between a solid and liquid form where the material undergoes a continuous transition in contrast to a first order transition such as melting ice. Using cellular automata as an example, he explores the effect of varying the rules for calculating the next state with a parameter, λ, which describes how likely it is for a cell to enter a *quiescent state*, a state where it will not disturb other cells until it leaves the quiescent state itself. At $\lambda = 0$ all cells enter a quiescent state after one step, representing a fully ordered system, thus at $\lambda = 0$ the system is in the ordered phase. At $\lambda = 1$ there are no rules that leads to a quiescent state, leading to very chaotic systems with very high entropy, representing the chaotic phase. As λ is increased from 0 the cellular systems starts to form intricate structures, increasing in complexity and taking longer to reach a steady state. This behavior peaks at $\lambda \approx 0.5$ which is the most critical phase in this particular system, creating complex self-organizing structures. As λ is increased further the self-organizing structures start to give way to more chaotic and seemingly random behavior, gradually transitioning into the chaotic phase.

It turns out then, that with most cellular systems providing the necessary conditions for computation to occur is a tractable problem, however harnessing and shaping this computation is a whole different matter. In their critical phase systems are unstable but not chaotic which causes them to be highly sensitive to small changes in topology and initial conditions. Attempting to maneuver this fitness-landscape by directly changing the properties of a system would in many ways be like calculating the correct way for a butterfly in china to flap its wings in order to cause a hurricane in New York.

C. Reservoir Computing

Faced with the intractability of designing cellular computation systems in a top down manner, and the unpredictable relation between cause and effect in self organizing systems has made it necessary to approach complex systems in a different manner. One such approach is to treat the system as a *reservoir* [5] which "acts as a complex nonlinear dynamic filter that transforms the input signals using a high-dimensional temporal map, not unlike the operation of an explicit, temporal kernel function."

AAAAAA 1 shows a typical RC (reservoir computing) system with three inputs and two outputs. The inputs are processed in a simple feed-forward neural network before perturbing the reservoir in some way. Similarly the state of the reservoir is being processed by an output layer before leaving the RC system. In the figure the input and output processing is done by feed forward neural networks, but we note that this is only one of many possible filters. Inputs 1, 2 and 3 are snapshots of the current state of the problem we attempt to solve with reservoir computing. Since our filters have no state, at least not beyond some time horizon we see that the history of the system must in some way be encoded in the reservoir in order for the RC system to solve problems in scope wider than the limited amount of state that may be contained in the filters.

D. Neuron Computing

The human brain consist of 100 billion neurons, forming a vastly complex hierarchy of behavior, from the neuron-level interactions to the interaction between different parts of the brain such as the hippocampus and the frontal lobe. The basic building block of the brain is the neuron, which even by itself is a very complex system, not at all analogous to a single transistor or an artificial neuron. These neurons communicate with each others using electric and chemical signals in a complex interplay between neurotransmitters, voltage spikes and even rearranging their physical structures, growing new connections and letting unused connections wither. The complexities of neurons have been widely studied, but for this paper a cursory introduction is sufficient. We will only consider a generalized version of the neuron, but in our experiments a plethora of different neurons are used, although they all share the basic similarities described here. The anatomy of a neuron is shown in 9 and can roughly be divided into the following parts:

1) Soma: The main body of the neuron. While we will view neurons as simple network nodes it is important to note that the neuron is highly complex, it can blah blah

2) Dendrites: To sense its surroundings the neuron is equipped with dendrites. These branching structures act as receivers, propagating electro-chemical stimuli to the cell body. Their reach is only to the immediate vicinity of the cell, they do not form longer connections.

3) Axon: The axon is a long tendril, extending over a meter in the case of the sciatic nerve, which transmits information as electrical pulses to other neurons. An axon can branch off and reach multiple neurons, it is not a one to one connection.

E. Adaptivity

Write about evolvable systems. Possibly evo-devo?

III. A Hybrid Neuro-Digital Approach

1) Concept: The concept of the cyborg is inspired by similar efforts such as [6] and [**?**]. Fig 2 shows the basic premise: An in-vitro neuron culture will be used to control a robot in a closed loop system, making the robot an actual cyborg.

2) Platform: The cyborg platform

3) Growing NIV:

4) A First Test:

A. MEA2100

To perform experiments a MEA2100 system has been purchased from multichannel systems. The MEA2100 system is built to conduct experiments on in-vitro cell cultures, with the main focus being on neurons. The principal components of the MEA2100 systems are:

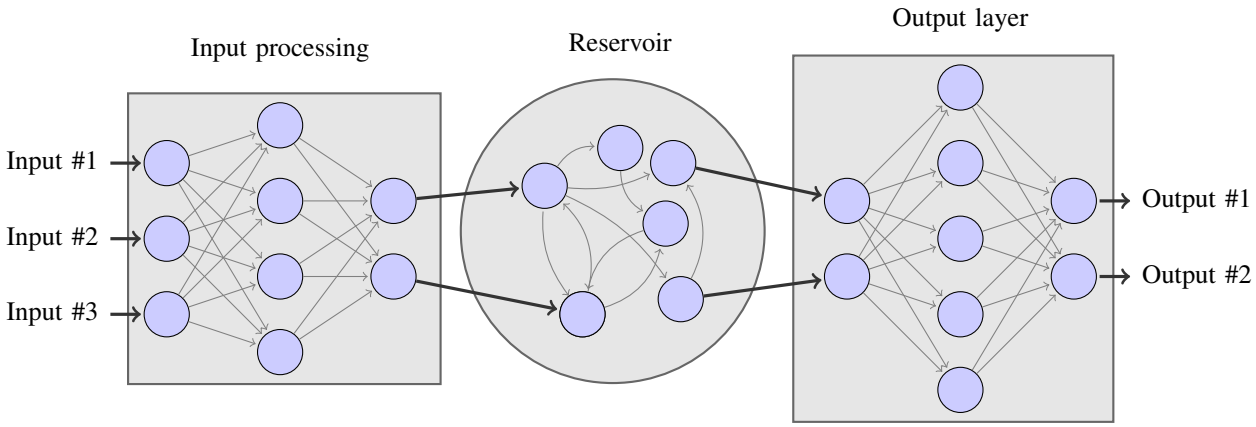

Fig. 1. A reservoir comput-thingy

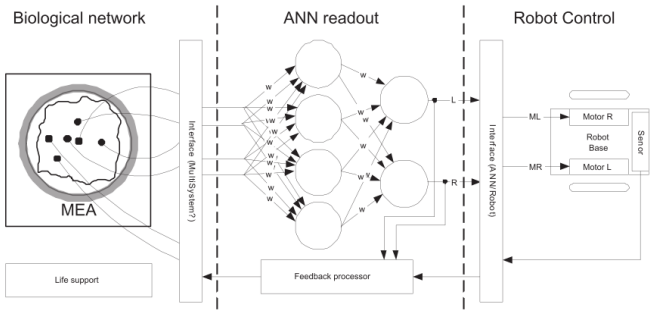

Fig. 2. The gist of it..

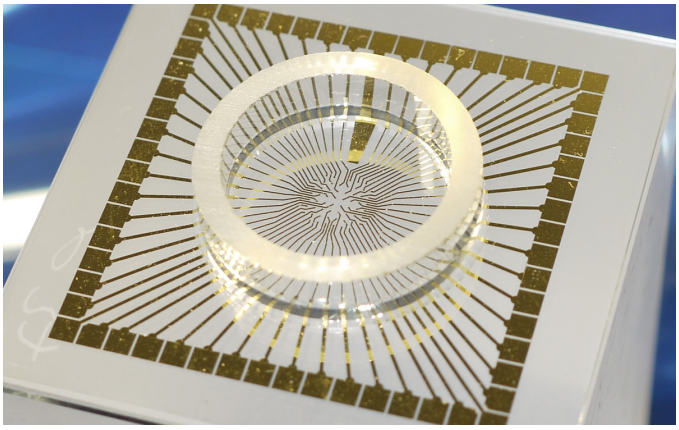

Fig. 3. A generic MEA

1) Micro electrode array: To experiment on the properties of cells or other electrically active subjects the micro electrode array (MEA) is used. As the name implies the MEA is equipped with an array of electrodes able to both measure the electrical properties of the experiment subjects, as well as applying outside stimuli, acting in a sense as output and input for the subject. 3 shows an empty MEA, **??** shows an MEA from st.olavs with a live neuron culture.

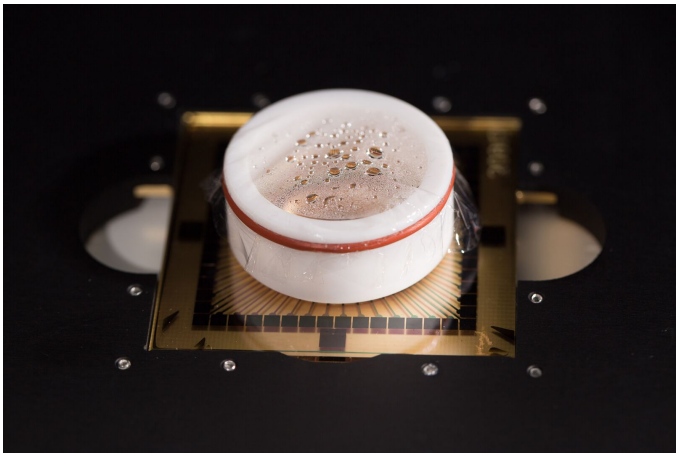

Fig. 4. A MEA with a live culture, photographed by Kai

2) Headstage: The electrodes of the MEAs are measured and stimulated by the headstage which contains the necessary high precision electronics needed for microvolt range readings. 5 shows the same type of headstage used in this paper along with an MEA.

3) Interface board: The interface board connects to up to two head-stages and is responsible for interfacing with the data acquisition computer, as well as auxiliary equipment such as temperature controls. The interface board has two modes of operation. In the first mode the interface board processes and filters data from up to two headstages as shown in 7 which can then be acquired on a normal computer connected via USB. In the second mode of operation a Texas instruments TMS320C6454 digital signal processor is activated which can then be interfaced with using the secondary USB port as shown in **??**

IV. Methodology

TO-DO: differentiate RC-agnostic post processing and spike detection. Should probably be done in background

Fig. 5. The headstage

Fig. 6. The MCS interface board

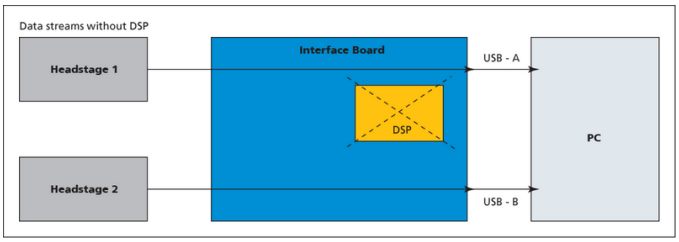

Fig. 7. Casual mode

The architecture described in this paper is a refinement of the neuro-robot architecture used in [6]. In [6] a system for working specifically with MEAs containing dissociated neurons is described with a TCP/IP connection to a slave PC controlling a robot being the main selling point. Our model is a generalization of [6] designed for reservoir computing, where

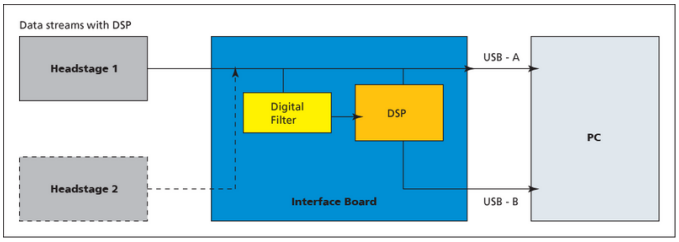

Fig. 8. DSP active

the neuron-culture is just one of many possible reservoirs. It should be noted that our architecture does not require experiments to be posed in the context of reservoir computing. (not too happy with the words here, make them betterer. Also something about the possibility of multiple implementations)

A. *Areas Of Concern*

The architecture has been defined by three areas of concern. This modularization facilitates reuse of code ??? some words ??? In this section it is important to note that we differentiate between generic data-processing and task specific processing. The former describes getting spike trains while the latter describes reservoir computing specific processing This is good because it separates RC from other stuff

1) Data Acquisition and Interfacing: Data acquisition entails configuring the MEA2100, collecting data from the MEA2100 and setting up stimuli. This means the data acquisition and interfacing software must at least be able to deliver a raw, that is unprocessed data-stream, as well as being able to accept requests for stimuli over TCP/IP. In practice the data acquisition software may also handle processing of acquired data for performance reasons, but it is not a required task.

2) Data Processing: This part of the architecture is responsible for processing data in a task specific, i.e non-generic way. The data may be raw waveform data or spike data from the MEA that needs to be processed into commands for an agent, or it may be sensor data from a simulated agent that must be processed into stimulus requests for the MEA. As with the data acquisition and interfacing module, the data processing module communicates over TCP/IP.

3) Agent Control: This module implements the actual embodiment of the neuron-culture. The agent control reads the processed data from the data processing modules and issues commands to an agent. It also transmits sensor data back to the data processing module, providing feedback to the neuron culture. The type of the agent is not specified, it can be a fully fledged walkin' talkin' cyborg, or it can be a simple simulated agent.

B. *Implementation*

Currently the following software systems have been implemented:

1) MEAME: MEAME implements the data acquisition and interfacing requirement of the closed loop system. It is written in C# and interfaces with an API provided by multichannel systems which allows it to interface with the MCS interface board. MEAME also implements the optional DSP which is used to stimulate the neurons thus fulfilling both the requirements.

2) SHODAN: SHODAN is a framework for composing reservoir computing experiments written in scala. In contrast to MEAME which is written specifically for interfacing with MEA2100, SHODAN is intended to facilitate general purpose input and output processing for reservoir computing. SHODAN currently hosts a number of tools for composing and reusing data-processing pipelines such as parametrizable feed forward neural nets, serialization and deserialization methods for TCP streams for communicating with the data-acquisition module and the agent control module. Additionally SHODAN hosts a simple simulator intended to stand in for a more full-fledged agent control module. This simple simulator simulates a simple agent with four eyes which reacts to proximity to a wall, and can visualized in a browser in real-time to get that sweet grant $$$

C. Interfacing With the MEA2100

V. Experiments

In order to show in vitro neuron cultures being viable reservoirs we have constructed a very simple simulated environment where an agent is placed in a 2D top down environment. The goal of this simulated game is to avoid walls while exploring (that is, covering ground in order to punish agents that run in circles or stand still). In this experiment we wish to show that the neuron reservoir enhances the performance

VI. Results

Here's where we add the results

VII. Conclusion

We conclude that this was an excellent idea but it would be more fruitful to investigate distributed webscale apps.

VIII. Further Work

For instance investigating growth rules for neurons in chaos or orderly environments to investigate if they trend towards a critical phase.

Acknowledgment

The authors would like to thank...
Sandoz methylphenidate 54mg, couldn't have done it without you.

References

[1] G. Tufte, "From Evo to EvoDevo: Mapping and Adaptation in Artificial Development,"

[2] N. Bertschinger and T. Natschlger, "Real-time computation at the edge of chaos in recurrent neural networks," vol. 16, no. 7, pp. 1413–1436.

[3] C. Gershenson, "Introduction to Random Boolean Networks,"

[4] C. G. Langton, "Computation at the edge of chaos: Phase transitions and emergent computation," vol. 42, no. 1, pp. 12–37.

[5] B. Schrauwen, D. Verstraeten, and J. M. V. Campenhout, "An overview of reservoir computing: Theory, applications and implementations," in *ResearchGate*, pp. 471–482.

[6] Y. Li, R. Sun, B. Zhang, Y. Wang, and H. Li, "Application of Hierarchical Dissociated Neural Network in Closed-Loop Hybrid System Integrating Biological and Mechanical Intelligence," vol. 10, no. 5, p. e0127452.

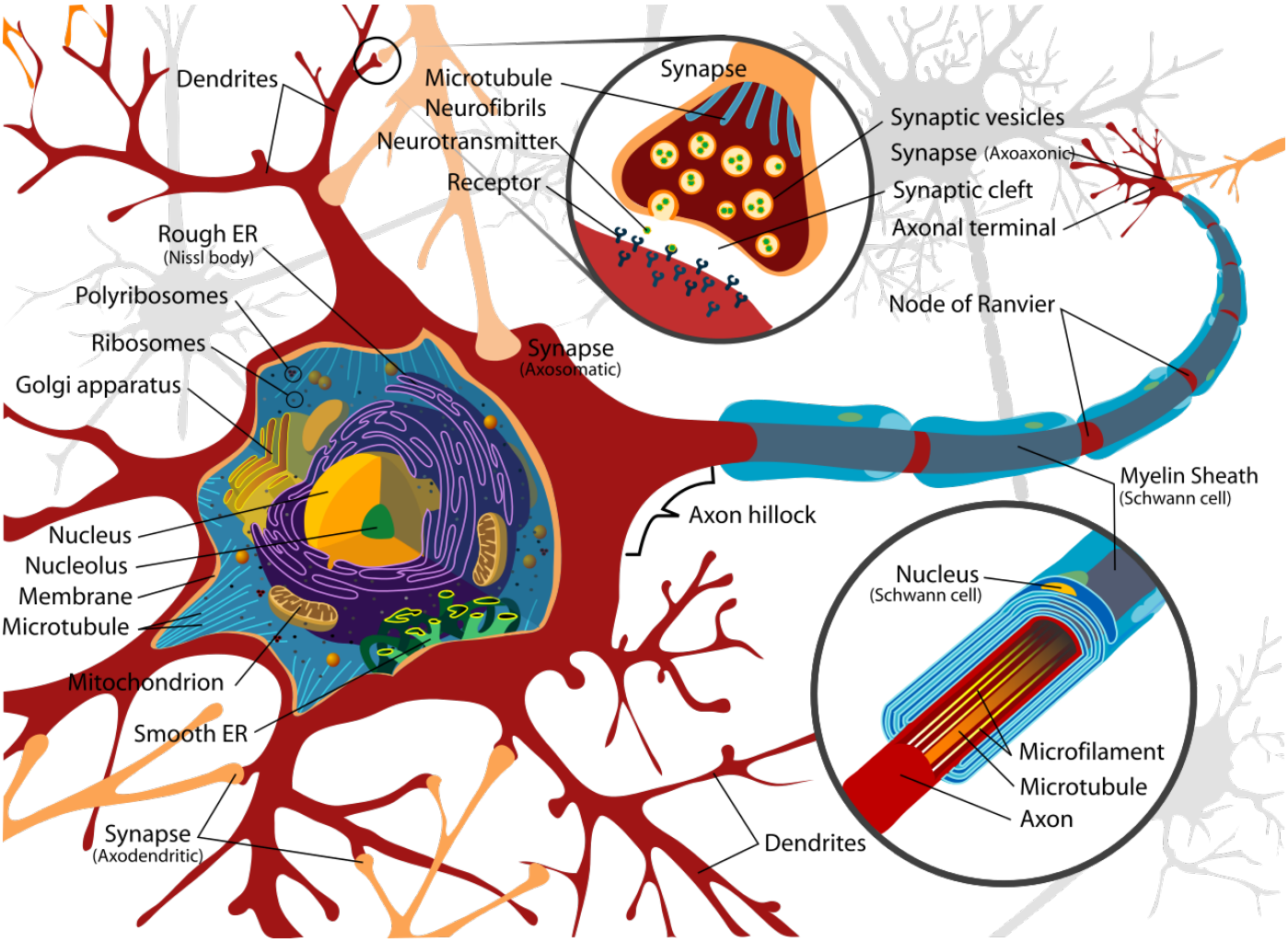

Fig. 9. a neuron