Inhaltsverzeichnis

1 Intro

1.1 Requirements

- Matlab (with Communications Toolbox)
- Putty / SSH Client
- 09:00 to (no later than) 17:00

1.2 Overview

1.2.1 Embodied AI

2 Biological-neurons

2.1 Neurons

- **Dendrite(s)**: Input(s)
- **Axion**: Output
- **Soma**: Cell body
- **Nucleus**: Cell core

Neurons collect electrical signals to process and transmit to other *neurons*. *Axon* terminals of one connect to *dendrites* of other *neurons*. *Synapses* are structures to connect those electrically/chemically (however no physical connetion is made).

2.2 Synapse

Electrical signal trasnission through Ion-filled Substrate.

- **Pre***synamptic neuron*: Sending Signals from Axiom
- **Post***synamptic neuron*: Receiving Signals at the Dendrite

Voltage changes open Voltage gates from the neural-fluid into the *presynaptic neuron*. This pulls in Ions from the neuralfluid maing *vesicles* release realese *neurotransmitteres* into the *synaptic cleft* to move between the *neurons*. The *postsynaptic neuron* receives these trasmitteres into receivers and converts the chemical information to an electrical signal.

2.3 Signals

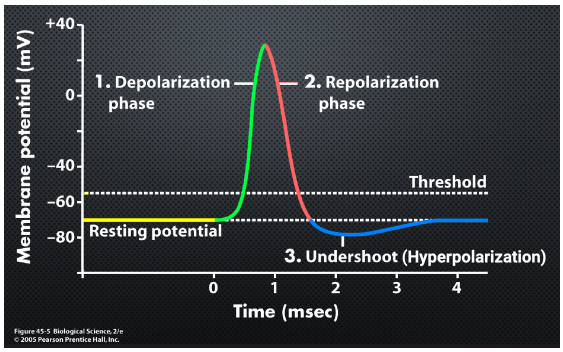

Abbildung 1: Neuron Spike

- *Resting* at -70mV
- *Depolarization phase*: Excitation from input signal reaching an artificial threshold resulting in a voltage jump (up to $+40\text{mV}$)

- *Repolarization phase*: Return to resting potential
- *Undershoot (Hyperpolarization)* return to resting.

All voltages with respect to outside brainfluid.

This process thakes around 3ms (333.33Hz). The *Myelin Sheaths* decreases performance and throughput aswell. This is faster due to thight packing and parallel processing.

The spike is seemingly identical between different neurons (same Amplitude and timeframe).

2.3.1 Resting membrane potential (default state)

Different ion concentration: more negative inside the neuron than outside. Measurement in reference to outside. Outside: Mostly $Na+$ and $Cl-$. Inside: $K+$ and $A-$. Resting voltage sits at around -65mV to -70mV.

2.3.2 De-polarization

Ions flow through the neuron. Signal excites gates. Gates are ion-specific and only allow certain kinds of ions. These are *voltage-gates channels*.

Ions like Sodium ($Na+$) enter the neuron resulting in a positive voltage swing up to $+40\text{mV}$. Once all Sodium gates are open the threshold is reached. The gates open with very little voltage.

2.3.3 Hyper-polarization

Once the voltage between neuron and outside fluid is positive, the Sodium gates close (as they're voltage controlled). Respectively the Potassium ($K+$) gates open. Positive charge leaves the neuron making the voltage drop to below the threshold. At resting potential the Potassium gates close. Due to the delay in closure undershoot occurs.

2.3.4 Encoding Information

Information is seemingly encoded in timeing between pulses. Amplitudes and Durations of spikes are too simmilar between spikes.

2.4 Artifical Neruons

2.4.1 Perceptrons

A simple neural model.

All dendrites u_i get weighted w_i and summed resulting in the activation z. The summation simulates the *soma* core.

$$z = \sum_{i=1}^{n} \omega_i \cdot u_i$$

The threshold and axiom are simulated by an activation function ϕ resulting in the *perceptrons*' output v.

$$v = \phi(z)$$

Activation functions tend to clamp the output in the range of -1 to 1.

An activation function dictates the output space. A heaviside function can only output a binary result. Functions with infinite range may diverge. Sigmoid functions can't overflow however the may saturate. The computataional cost is quite prohibitive.

2.4.2 Hodgkin-Huxley Model

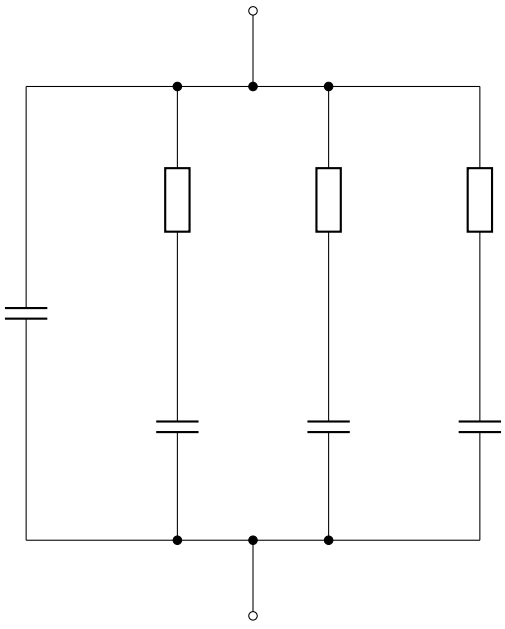

Abbildung 2:

3 Artificial neural brains

3.1 Braitenberg Vehicles

- **Ipsilateral**: Connections on same side
- **Contralateral**: Connections cross sides
- **Excitatory**: Input Increases → Output Increases
- **Inhibitory**: Input Increases → Output Decreases

Vehicle emulates simple P-type control.

Mathematical model includes:

- s_x: Sensor value
- v_x: Output value
- k: Linear proportional gain

Mathematical example implementations:

- **Ipsilateral**: $v_{\text{left}} \propto s_{\text{left}}$
- **Contralateral**: $v_{\text{left}} \propto s_{\text{right}}$
- **Excitatory**: $v \propto s$
- **Inhibitory**: $v \propto \frac{1}{s}$

Pathplanning: Finding path from known start to known end including known obstacles.

Complex behavior emerges by combining multiple weighted control loops running in parallel.

4 Artificial learning

4.1 Plasticity

Neuroplasticity: Ability for the brain to re-organize itself in both *structure* and *function* over time due to external and internal events. **Neuroplasticity** is mechanism behind "*learning*" and is happening continuously.

Structural Plasticity	**Functional Plasticity**
new neural connections	changing existing connections
long-term changes	short term changes

Plasticity happens on all levels from cortical down to the synaptic level.

- **cortical**: changing stimulus from limbs triggers different existing neurons
- **synaptic**: changing amount of gates on post-synaptic neurons' dendrites

4.2 synaptic strength in functional plasticity

4.2.1 Long Term Potentiation (*LTP*)

HFS: 100 Pulses (over 1s $\rightarrow$ 100Hz) as an input to a neuron. The neuron is resting at $t = 0$. The **HFS** hits the neuron resulting in an instantaneous output, the **LTP**. The neurons output jumps, then receedes and continues to saturate (Only as long as the **HFS** is continous.) The **synaptic strength** is the chance the output is increased.

A lot of fast input $\rightarrow$ Big changes and high learning

LTP increases synaptic strength

4.2.2 Long Term Depressino (*LTD*)

The Inverse, to decrease the **synaptic strength** an **LFS** (900 Pulses 15min $\rightarrow$ 1Hz) is sent. The neuron responsds, dips and saturates in a depression.

Low data $\rightarrow$ Low learning

LTD decreases synaptic stregth

4.2.3 Chemical basis

LTP and **LTD** result in synapeses by creating or destroying gates at the pos-synaptic terminal respectively.

4.3 Hebbian learning model

Efficiency describes the likelyhood if a presynaptic neuron spiking and exciting it's postsynaptic neuron. The likelyhood of the post-synaptic neuron firing after having been exicted is increased. More firing together $\rightarrow$ more likely to fire together in the future. They spiking is, however, *not necessarily causal*. At high efficiency the spiking of both neurons are **temporally correlated**. The spiking is **associative** and **unsupervised**.

Neurons that fire together, wire together.

4.3.1 Simple mathematical model

$$\frac{d\omega_1}{dt} = \mu \cdot v \cdot u_1$$

- ω: dsecribes the synaptic strength / weight
- $\frac{d\omega_1}{dt}$: (not a derivative), Change in synaptic weight
- μ: Learnig rate ($\mu \ll 1$ to avoid "exploding learning problem")
- v: Output of post-synaptic neuron
- u_1: Output of pre-synaptic neuron / input to post-synaptic neuron

$$\omega_n = \omega_{n-1} + \frac{d\omega_{n-1}}{dt} = \omega_{n-1} + \mu \cdot v \cdot u_{n-1}$$

Problem: ω_1 is always increasing, unstable ~~but biologically correct~~. This is an open control loop.

As this is unsupervised we don't have an error term and can't simply stop when the model is "good enough".

4.3.2 LTP

The further the amount of time between two spikes firing the more the weight changes. A high δt results in little change, a small δt results in large changes. At $\delta t = 0$ maximal change occurs. The simple model only results in positive change, thus unstable.

4.4 Input correlation learning (*ICO*)

Learning rule

$$\frac{\delta w_a}{\delta t} = \eta \cdot f\left(A, t\right) \otimes \frac{\delta f\left(B, t\right)}{\delta t}$$

- η: learning rate
- $f\left(A, t\right) \otimes \frac{\delta f(B,t)}{\delta t}$: Temporal correlation
- $otimes$: cross correlation
- A: Predictive signal
- B: Reflex signal
- Y: Neuron Ouput
- w_a weight between A and Y
- f output function of a neuron (including the sigmoid)

If we'd like to stop the learning we can assume B to be constant. We cannot guarantee $B \to 0$ (to stop learning) but we can take the derivative to stop learning once stimulus ceases change.

This Algorithm **will converge** to the correct weight.

Output signal is the weighted sum.

$$Y(t) = w_a \cdot f\left(A, t\right) + f\left(B, t\right)$$

4.4.1 Perceptron learning

Learning by updating input weights only. Update done using **gradient descent**.

Update weight in proportion to contribution to the output. Contribution is the change in error E für a given change in w, where the mean squared error is defined as

$$E = \frac{1}{2}\left(t - v\right)^2$$

- t: target output
- v: actual output

Determining error requires a known correct output.

$\to$ **supervised learning**

5 Supervised & unsupervised learning

5.1 Non-linear actiavtion function

Bias is activation function x-Offset **Slope** of activation function is rarely used.

An example sigmoid with **bias** and **slope**.

$$v = \frac{1}{1 + e^{-S(z-b)}}$$

where b is **bias** and S the **slope**. **Bias** can be used as a weight.

$$\begin{aligned} z &= w_1u_1 + w_2u_2 + \ldots + w_nu_n \\ &\rightarrow (z - b) = w_1u_1 + w_2u_2 + \ldots + w_nu_n - b \\ &\rightarrow (z - b) = w_1u_1 + w_2u_2 + \ldots + w_nu_n - (b \cdot -1) \end{aligned}$$

$\rightarrow (b \cdot -1)$ splits to w_{n+1} and u_{n+1}

This results in an additional weighted bias shifting the activation function resulting in

$$z = \sum_{i=1}^{n+1} \omega_i \cdot u_i$$

Each perceptron can implement one **deciscion boundary**. **Deciscion boundaries** seperate inputs into different classes. The boundary can be shifted by adapting the weights.

By adding more perceptrons the **deciscion boundaries** dimension increases. The boundry of 2 neurons results in one-dimensions. 3 Neurons create a 2-Dimensional **deciscion boundary**. More Neurons build more complex spaces.

5.2 Designing a network

- Defined number of inputs
- Defined number of outputs
- Variable hidden layers

Hidden layer depends on linearity of the problem. No general solution to amount of hidden layers. Strategy of trial and error, start with ≈ 100 layers.

Deep Neural Networks: Depth is defined horizontally.

5.3 Convolutional neural networks

Hereby:

- u: Input
- v Output
- x Hidden layer
- w Weight from u to x
- y Weight from x to v

One **Iteration** consists of one forward pass and one backwards pass. One **Epoch** consists of **Iterations** for all Items in the training set.

5.3.1 Forward propagation

1. Set input
2. Calculate for all hidden layers
$$x_j = \sum_i u_i w_{ji}$$
3. Calculate for all output layers
$$v_k = \sum_j x_j w_{kj}$$

5.3.2 Backpropagation

1. Calculate error gradient for all output neurons
$$E_k^0 = v_k(1 - v_k)(t_k - v_k)$$
2. Calculate error gradient for hidden layers
$$E_j^h = x_j(1 - x_j) \sum_k E_K^0 y_{kj}$$
3. Update weights for outputs
$$y\prime_{kj} = y_{kj} + \mu E_k^0 x_j$$
4. Update weights for hidden layers
$$w\prime_{ji} = w_{ji} + \mu E_j^h u_i$$

5.4 Vanishing Gradients

With a great amount of layers the impact of early neurons (close to the input) have less effect on the output error and get changed less resulting in less learning. A high amount of layers does not guarantee better network performance.

Dropout randomly disables neurons and stops updating their weights. This does not guarantee better accuracy only better execution speed.

This only occurs by learning with backpropagation.

Alternative: **NEAT** (*Neuroevolution of augmenting topologies*) using generative algorithms. Possibly (not guaranteed) better performance to optimize output by chaning the entire networks structure. Worst execution speed and memory performance.

5.5 Trainig proceedure

Split trainig dataset into two parts to avoid overfitting. Suggested split:

- 70% training data
- 30% testing data

Initilize weights to random values.

- **Training Datasset**: Adjust weights/learn
- **Testing Dataset**: Testing final solution
- **Validation Dataset**: Minimize overfitting

Always randomize oder of data for every **epoch**, as netoworks easily learn patterns.

More complex splitting algorithms and procesdures:

- **Monte Carlo corss validation** subsamples data randomly into its sets.
- **K-fold corss validataion** divides data into k subsets, trainig it and removing it after training to repeat with the remaining $k-1$ subsets
- **Leave-p-out cross validation** p datasamples, use $n-p$ for training, but test and train $\frac{n!}{p!\cdot(n-p)!}$ times. This presents every datapoint equally often and fairly.

6 Questions & Answers

6.1 Biologically inspired robotics

1. *What is biologically-inspired robotics?*

   Using biological systems as inspiration for robotic design.

2. *What is biorobotics?*

   Using the biologically inspired robotic system to better understand the original biological system.

3. *What is the difference between biorobotics and biologically-inspired robotics?*

   Biorobotics combines existing biological systems with mechanical systems. Biologically-inspired robotics takes inspiration of biological systems for robotics without combining the two. **(?)**

4. *What is the reality check in biorobotics?*

   Complex biological behaviour doesn't arise from complex systems but rather simple, yet dedicated, systems.

5. *What is neurorobotics?*

   The study and application of science and technology of embodied, autonomous, brain-inspired algorithms.

6. *What is a neurorobot? Can you give an example of a neurorobot?*

   A breitenberg vehicle.

7. *What is embodied AI?*

   A purpose built mechanical system in conjunction with an AI controller.

8. *Can you explain the three-layer embodied AI architecture?*

   - The controller acting as a supervisor: The brain
   - The mechanical input and outputs: Motors and Sensor
   - The environment

9. *Can you give an example of embodied AI in humans/animals? Can you give an example of embodied AI in robots?*

   Gas detecting robot inspired by cockroaches.

10. *Why is the environment important in embodied AI?*

    An enviornment influences an agent. An agent must overcome external influences or take advantage of them. Example: Seagulls in hovering in gusts of wind without effort.

11. *What are model-based and model-free approaches in biorobotics? Can you use both together in one robot? Can you give an example?*

    - **Model based**: A mathematical model defines the entire robotic system.
    - **Model free**: The robot acts according to some defined rules but "figures out" how to achive it's objective by itself.

    Both appoaches can be used together. A model-based approach defining the systems baseline behviour with the model-free system adapting to external changes.

    Examples: 3-D Printer controlling it's nozzle temperatur with a model (PID Controller) and minimizing motor vibrations using a learning algorithm.

6.2 Neurons

1. *What is a neuron?*

   A biological braincell responsible for processing information.

   It consists of multiple Inputs (**Dendrites**), one Output (**Axiom**) and it's **soma** with **nucleus.

2. *What are the different types of neurons?*

   - **Unipolar**: Inputs, from outside the brain (found in insects)
   - **Bipolar**: Inputs, from senses (eyes, ears)
   - **Multipolar**: Within the brain and as outputs to muscles
     - **Pyramidal**: complex though (Cerebrum)
     - **Purkinje**: reactive action (Cerebellum)

3. *How does a neuron work/transmit a signal through itself?*

   Using electrical spikes, presumeably through the spike signal's frequency and timing charachteristics, not by it's shape, amplitude or phase. **(?)**

4. *How does a neuron forward a signal to other neuron(s)?*

   By using chemical reactions at synapses.

5. *What is a synapse?*

   A non-physical connection between two neurons.

6. *How does a synapse transmit a signal?*

   The electric signal in the pre-synaptic neuron releases neurotransmitters into the synaptic chasm. These travel through the brain fluid to the post-synaptic neurons neuroreceptors. If enough receptors get stimulated they create an electric spike.

7. *What is an action potential?*

   An electrical signal in a neuron.

8. *Describe how an action potential is generated.*

   By the neuroreceptors in the post-synaptic neuron.

9. *Describe how the different phases of the action potential are generated.*

   - The **resting phase** is the default state.
   - The **depolarization phase** occurs once the Natrium ($Na+$) gates reach a voltage-threshold. A voltage overshoot occurs as reactions are not instantenous.
   - The **repolarization phase** follows showing a decline in voltage as the Sodium gates saturate and repell ions, thus allowing Potassium ($K+$) to enter.
   - The **hyperpolarization phase** returns to the resting state as the voltage reaches an equilibrium.

10. *What is resting potential/depolarisation/repolarisation/hyperpolarisation? What mechanism(s) causes each phase to be generated?*

    See above answer.

11. *Why is the resting potential negative?*

    As ions leak through the neurons membrane the neuron takes on a more negative charge in respect to the outside fluid.

12. *How is information encoded in spikes?*

    Presumeably in its frequency and timing charachteristics.

13. *What is a perceptron?*

    A quantized model of a neurons behavior. Consisting of multiple weighted inputs getting summed and passed through an activation function.

14. *What is an activation?*

    The activation describes the weighted sum of the perceptrons inputs. $z = \sum_{i=1}^{n} \omega_i \cdot u_i$

15. *What is an activation function? Why do we need it/What will happen if I don't use it?*

    A mathematical function limiting the perceptrons activation (weighted sum) to a knwon output space.

16. *What are the different types of activation functions and their advantages/drawbacks? Which one will you choose and why?*

$$\text{RelU} = \begin{cases} 0 \text{ for } x < 0 \\ x \text{ for } x \geq 0 \end{cases}$$

$$\text{Heaviside} = \begin{cases} 0 \text{ for } x < 0 \\ 0.5 \text{ for } x = 0 \\ 1 \text{ for } x > 0 \end{cases}$$

$$\text{Linear} = x$$

$$\text{Sigmoid} = \frac{1}{1 + e^{-x}}$$

- Heaviside limits the output to a binary state, severly reducing granularity.
- Linear and RelU are computationally efficient but may grow unbounded.
- Sigmoids limit the effective range and don't quantize dramatically.

17. *Why do we use a sigmoid activation function (think about it from both computer implementation and biology perspectives)?*

    It binds the output to a manageable range which avoids overflow and offers great precicion using IEE754. It also models neurons' own saturation.

18. *What is the basic difference between biological neural processing and artificial neural processing?*

    Biological neural processing can operate many billions (or more) neurons in parallel. Computing with artifical neurons cannot currently achieve this.

19. *What is a Hodgekin-Huxley model? What similarities/dissimilarities does it have with a biological neuron? What drawbacks does it have in terms of implementation?*

    An electronic circuit aiming to reproduce neurons' voltage spikes. It models different Ion-charges and resistances using batteries and resistors acting upon a cpacitor and receiving an external input.

    The model only offers one input. It breaks down with too-high currents failing to simulate accordingly. The model only generates spikes and it's output is not generally useful for computing.

    It's implementation is computationally incredibly expensive requring mutliple equations to be evaluated for every timestep of the voltage signal.

6.3 Braitenberg vehicles

1. *What is a Braitenberg vehicle? Can you give an example of one and how it behaves?*

   A simple concept of a neurorobot consisting of two powered wheels receiving input from two sensors placed at the top left and right of the vehicle. The connections between the sensors and wheels dictate the vehicles emerging behaviour.

   An agressive vehicle features **contralateral** and **excitatory** connetions and agressively manouvers towards the sensor's gratests stimulus. The „love"vehicle consists of **ipsilateral** and **inhibitory** connections resulting in it seeking the source of gratests stimulus but slowly coming to a halt when approaching it.

2. *Are there any advantages of using Braitenberg vehicles?*

   Their rather complex behavior emerging from very simplistic rules make them computationally trivial.

   They enable emulating behaviors of simple insects.

6.4 Learning

1. *What is neuroplasticity? What types of neuroplasticity are there?*

   Neuroplasticity describes the ability of the brain to re-organize itself over time in order to learn. There is **structural** and **functional** neuroplasticity.

2. *What is structural/functional plasticity? What are the differences between the two?*

   - **Structural**: Major structural changes over long periods of time
   - **Functional**: Minor adaptions over short periods of time

   There are more biological mechanisms for **structural** than for **functional** changes.

3. *Can you give an example of structural/functional plasticity?*

   - **Structural**:
     - Synapses changing in number
     - Synapses receptors changing in density
   - **Functional**:
     - Synapses adjusting their strength (*synaptic plasticity*)

4. *What is Long Term Potentiation? What is Long Term Depression? Can you say something about the chemical process and any changes underlying LTP and LDP?*

   LTP: If the synapse is overstimulated for a long period of time (many pulses in quick succession, high frequency) the synapse will create more neural receptors and thus „strengthen "the conection. The *fEPSP* slope increases.

LTD: If the synapse is, however, understimulated for a long time (very few pulses, low frequency), it will reduce the ammount of neuroreceptors at the post-synaptic neuron essentially „weakening"the conection. The *fEPSP* slope decreases.

(?)

5. *What is Hebbian learning? What are its drawbacks as a model for learning?*

   „Neurons that fire together, wire together "

   It describes the likelyhood of a presynaptic neuron spiking and exciting it's postsynaptic neuron. The likelyhood of the post-synaptic neuron firing after having been exicted is increased. More firing together $\rightarrow$ more likely to fire together in the future. They spiking is, however, **not necessarily causal**. At high efficiency the spiking of both neurons are **temporally correlated**. The spiking is **associative** and **unsupervised**.

   It's model is describes as

   $$\frac{\partial \omega_1}{\partial t} = \mu v u_1$$

   wherby the partial term $\frac{\partial \omega_1}{\partial t}$ may grow unbounded. As this control loop is unsupervised we can't stop the model once it's „good enough".

6. *What is Spike-Timing Dependent Plasticity?*

   STDP describes neurons' change in synaptic weight in relation to **LT Potentiation** and **LT Depression**. This proves a biological basis for Hebbian learning, as neurons which have a strong temporal correlation have a realtively strong synaptic weight.

7. *Do you know other forms of functional plasticity?*

   - **Homosynaptic** plasticity: changes in synapse strength occur only at post-synaptic targets that are specifically stimulated by a pre-synaptic target.
   - **Heterosynaptic** plasticity: activity of a third neuron can releases chemical neuromodulators that induce changes in synaptic strength between two other neurons.
   - **Non-synaptic** plasticity: intrinsic excitability, i.e. sensitivity to synaptic input, of neurons can be altered and is manifested as changes in the firing characteristics of the neuron itself.
   - **Homeostatic** plasticity: capacity of neurons to regulate their own excitability relative to network activity, a compensatory adjustment that occurs over the timescale of days.

8. *What type of plasticity is Hebbian learning (homo-, hetero-, non- or homeostatic)?*

   Hebbian learning is **homosynaptic** plasticity. **(?)**

9. *What is ICO learning? What type of synaptic plasticity is it and why? Can you give an example of how ICO learning can be used in robots?*

   Input correlation learning (**ICO**) is **heterosynaptic** plasticity.

The goal is to detect an event which triggers a reflex signal using a predictive signal (without even triggering the reflex). This is implemented using the correlation of these two (predictive and reflex signal). Implementing this requires a third neuron (one for each input and one additional neuron as output), thus **heterosynapsis** is required.

Example: Detecting obstacles using a camera (predictive input) without triggering the bump sensor (reflex input). The robot can learn to avoid obstacles by using the camera only.

10. *How do perceptrons learn? What is the fundamental difference between ICO learning and perceptron learning (think in terms of how gradient descent works vs how the ICO learning rule works)?*

    - **ICO** learns by adapting the time between predictive input and reflex input (the correlation between those). This process is **unsupervised**.
    - **perceptrons** learn by adjusting their weight by trying to minimize some given error function. This requires an expected output and is **supervised**.

11. *What is a Multi-Layer Perceptron (MLP)?*

    An **MLP** is a neural network. It consists of an input layer, an output layer and a defined amount of fully convoluted hidden layers inbetween those two.

12. *What is the backpropagation algorithm and how does it work? Why is it called a gradient descent method?*

    Backpropagation describes the process of changing weights in accordance to their respective impact on the output error. An errorgradient is calculated for each output neuron and for each hidden layer up to the input. Based on this gradient each weight is adjusted accordingly.

    The gradient describes an n-dimensional vector pointing towards the steepest decent on the error manifold's surface. Decending this is called gradient decent.

7 Robot

- `192.168.X.1`
- SSID = `BIOROLE2022_RobotX`
- PWD = `BioRoLe2022`

Login:

- User: `pi`
- PWD: `raspberry`