0.1 Neural networks

0.1.1 In silico neural networks

Artificial neural networks (referred to just as neural networks) is a software implementation of the connection of neurons in the brain. In computer science they can be used to solve a wide variety of problems, like character and facial recognition. They present an advantage over conventional programmatic methods, as they don't need explicit coding for each new problem. The simplest instance of neural networks is the single neuron, also known as the perceptron [1]. The perceptron can be used for simple decision making (giving yes/no answers) from a set of inputs. The logical 2-input AND operator is an example of a problem that can be solved with the perceptron [2], where the perceptron can output 1 if both the inputs are 1, otherwise 0.

The perceptron works by taking all the inputs, and multiplying each input by its associated weight. If the sum of the multiplied inputs exceeds a certain threshold, the perceptron will output 1, otherwise 0. This is shown schematically in Figure 1.

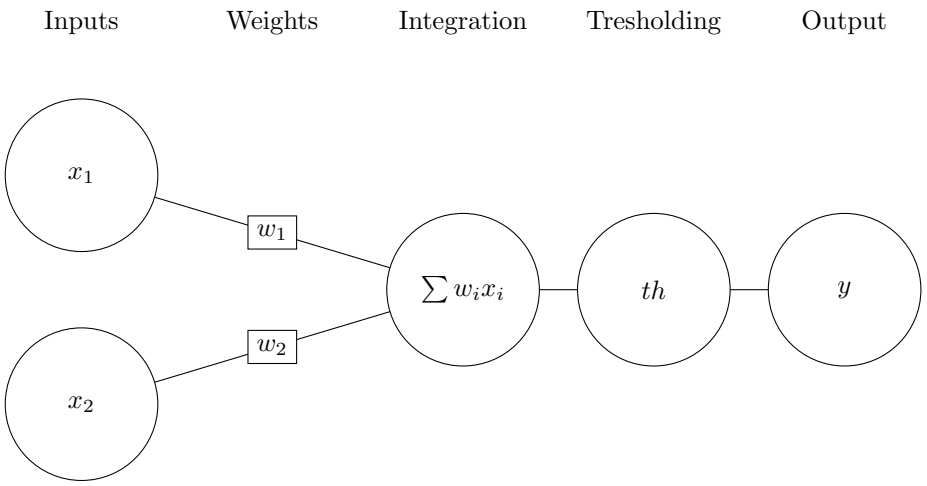


Figure 1: Diagram of the artificial perceptron.

The representation can be simplified to the type shown in Figure 2. The weights and thresholds for the perceptron version of the 2-input AND gate can also be seen in Figure 2.

The perceptron is limited to classifying inputs that are linearly separable [2], which rules out the XOR operation.

0.1.2 In vitro perceptron

It has previously been shown that the function of the artificial neuron can also be implemented using strand displacement reactions. The system is based on

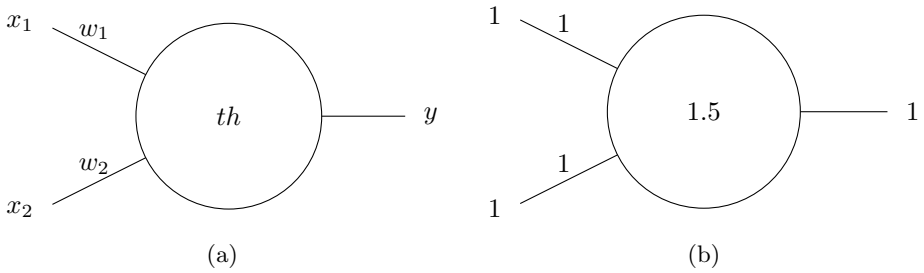


Figure 2: **(a)** Simple representation of a 2-input perceptron. **(a)** Weights and thresholds on a perceptron that implements the 2-input AND gate.

the seesaw gate motif [3], and can fullfill most of the functionality of a real neuron [4]. The seesaw gate is a catalytic gate with a threshold, designed for use in scalable strand displacement circuits. The seesaw gate uses a toehold to accelerate reactions, and has a left and right recognition domain to connect to other seesaw gates. The seesaw is named after the back and forth reaction of the strand displacement reactions, seen in Figure 3.

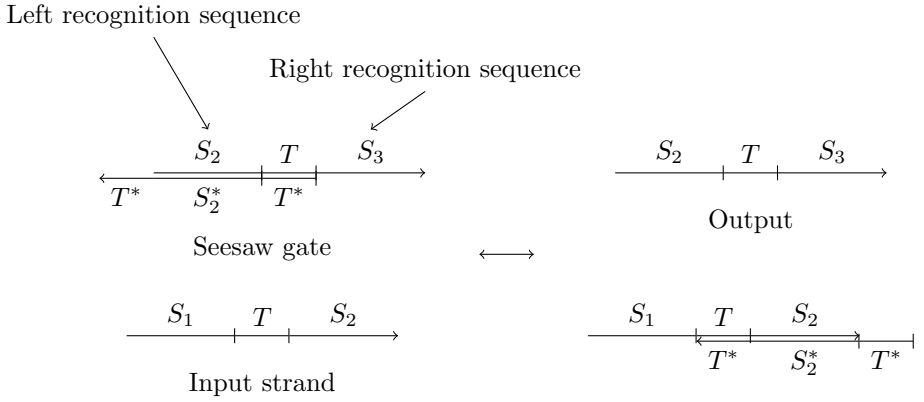


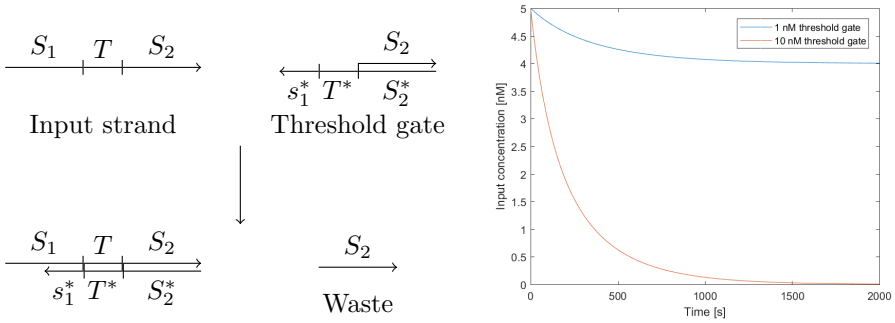
Figure 3: The back on forth reactions of the seesaw gate.

The reaction can be pushed to the right by using a fuel strand, or the input can be reduced by using a threshold gate, the details of which is discussed below.

Thresholding

In the natural neuron, the neuron will activate when its inputs exceeds a threshold. This is implemented using a threshold gate which will bind the input and prevent it from reacting downstream in the network. If the threshold gate concentration is higher than the input concentration, the input will be suppressed by the threshold. If the threshold gate concentration is lower than the input

concentration, not all of the input is suppressed, and will be able to react further downstream in the network.



(a) (b)

Figure 4: **(a)** Reaction of an input strand with a threshold gate. The product has no free toehold domain, and can't undergo reverse reaction. The waste has no toehold, and can't parcitipate in further reactions. **(b)** Time analysis of the concentration of input strand (5 nM start concentration). A threshold concentration higher than the input (red) will bind all input strand. A lower concentration (blue), will allow the input strand to participate in further displacement reactions.

Integration

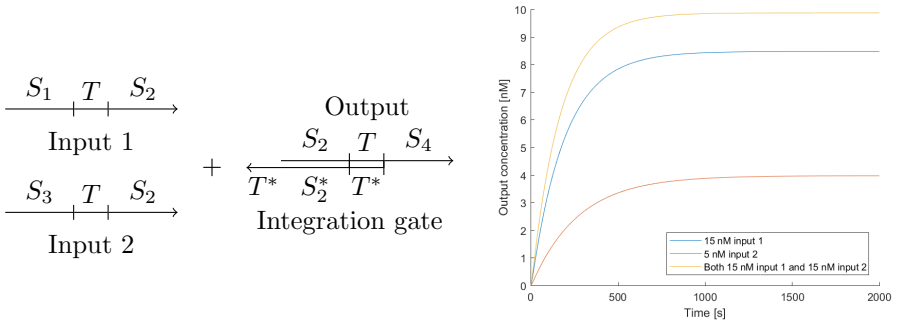
The input to the neuron have to be collected before thresholding, as they don't have the same left recognition sequence. This is done through an integrating gate, which will collect all inputs with the same right recognition sequence, and release a common signal which can be thresholded.

Weighting

The inputs to the gate are weighted using their concentration. By making the integration gates concentration the sum of all the input concentrations, a high concentration of one input strand will contribute more to the activation than that of a low concentration, as shown in Figure 5b.

The weight is decided by the concentration of output of a input neuron and its fuel strand. The fuel serves the purpose of pushing the output of one gate to its target concentration.

A problem with this approach to weighting, is that the inputs can only contribute positively to the sum. In silico neural networks can use negative weights to simulate inhibitory synaptic connections, and is needed to implement many kind of boolean functions. The in vitro network can't have negative concentrations of input sequences, so other approaches have to be considered. The problem can be solved using dual rail logic, where each input is replaced



(a) (b)

Figure 5: **(a)** Reaction of 2 input strands with an integration gate. The input strands have the same right recognition sequence S_2, and will both displace the top strand of the integration gate, releasing the output. **(b)** Time analysis of the concentration of the output strand. The concentration of the integration gate is 20 nM. The first input of 15 nM increases the output strand concentration, compared to the second input of 5 nM. When both strands are added the concentration of output increases further.

by two inputs. The details of the dual rail logic circuits is not a part of this report, and thus the networks will be limited to simple AND and OR circuits.

Neuron

By combining the discussed functional elements of the seesaw gate, a neuron can be created using strand displacement reactions, as depicted in Figure 7.

Instead of displaying all strands of the seesaw circuit, each gate can be represented by a simplified seesaw gate schematic. The first input gate from Figure 7 is shown as a schematic in Figure 8.

Using the schematic representation, the seesaw neuron from Figure 7 can be simplified to the schematic in Figure 9.

The recommendations from [4] is that each fuel strands concentrations is twice that of the gate concentration. The threshold concentration on the input gates (as shown in Figure 9) allows for concentrations of input strand less than 2 nM to still be detected as 0.

The representation from Figure 9 can further be simplified to the original perceptron shown in Figure 1. The seesaw neuron thus successfully implements all the required functionality of the artificial neuron, except for negative weights.

Training

The training algorithm typically used for digital perceptrons is based on the ability of the neurons to have negative weights and thresholds. A modified

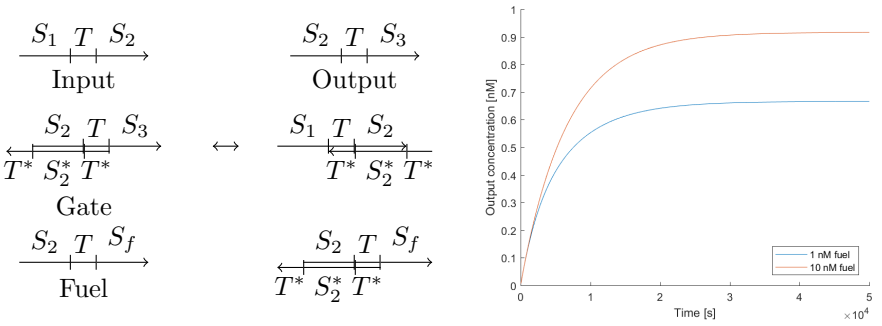


Figure 6: **(a)** Reaction of an input strand with a gate and fuel. The input displaces the output strand from the gate, and the fuel blocks the reverse reaction, pushing the equilibrium towards the free output. **(b)** Time analysis of the concentration of the output strand. Input and gate concentration is initially 1 nM. When the fuel concentration is low (blue), the output concentration doesn't reach the initial gate concentration. When the fuel concentration is high (red), the output concentration is pushed closer towards its maximum concentration.

Listing 1: Pseudocode for the seesaw perceptron training algorithm

```
initialize all weights to 0 and threshold to 10
while training is incomplete
  for all the input sets
    find the output of the network from the input set
    if output is not correct
      mark training as incomplete
  if training is marked as incomplete
    increase the weights
```

algorithm is presented in [4], which works for dual rail circuits. For the circuits in this report without dual rail logic, the algorithm can be seen in Code 1.

This process will keep increasing the weights until the output of the network matches the desired output. Without negative weights this will only work for AND and OR logic.

The input sets are taken from the truth table that the circuit should realize. If the circuit for example should realize the simple AND gate (see Table 1), the network is activated with the input rows from the truth table, and tested whether the output equals (or is very close) to the output in the truth table.

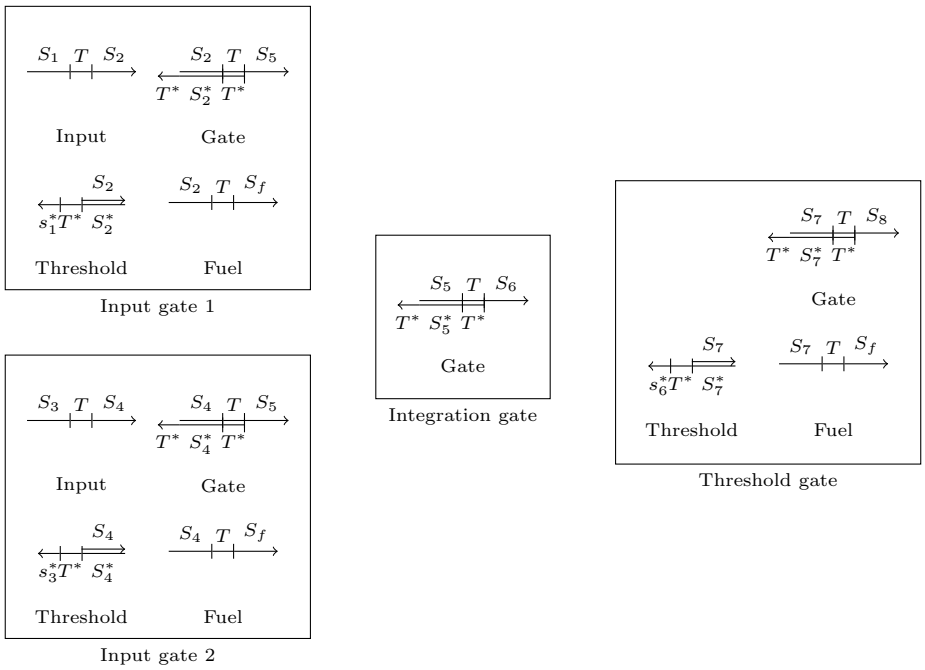
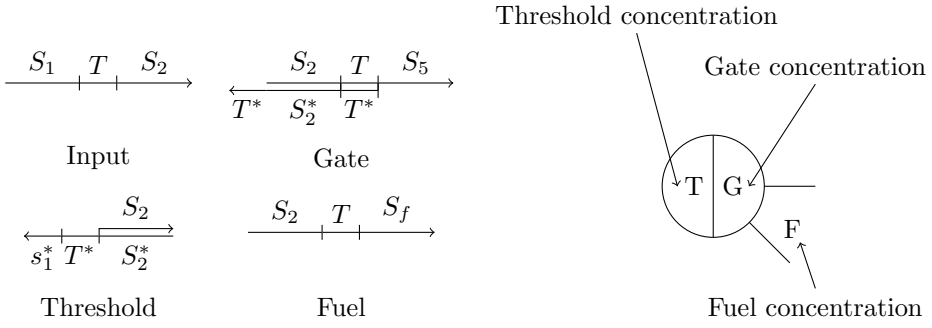


Figure 7: Schematic of a 2-input neuron implemented with seesaw gates. The neuron has an input gate for each input, which makes sure that the input is higher than a give threshold before the input is registered. The gate and fuel concentrations in each input gate affects the weight of the input before it is sent to the integration gate. The integration gate collects the right recognition sequence from the input gates for the threshold gate. The threshold gate activates if the sum of the weighted inputs is larger than the threshold concentration of the treshold gate.

0.1.3 Input translation

As per design of the seesaw gate, the input sequences for the strand displacement neural network needs very specific left-recognition, right-recognition and toehold sequences. To detect sequences that does not have these elements, the sequence has to be translated. It has previously been demonstrated that miRNAs can be translated to other sequences [5], by using two half-translators.



(a) Seesaw gate strands

(b) Seesaw gate schematic

Figure 8: The strands and concentrations of the units of a seesaw gate can be represented schematically to simplify larger diagrams.

Input 1	**Input 2**	**Output**
0	0	0
0	1	0
1	0	0
1	1	1

Table 1: Truth table for an AND gate.

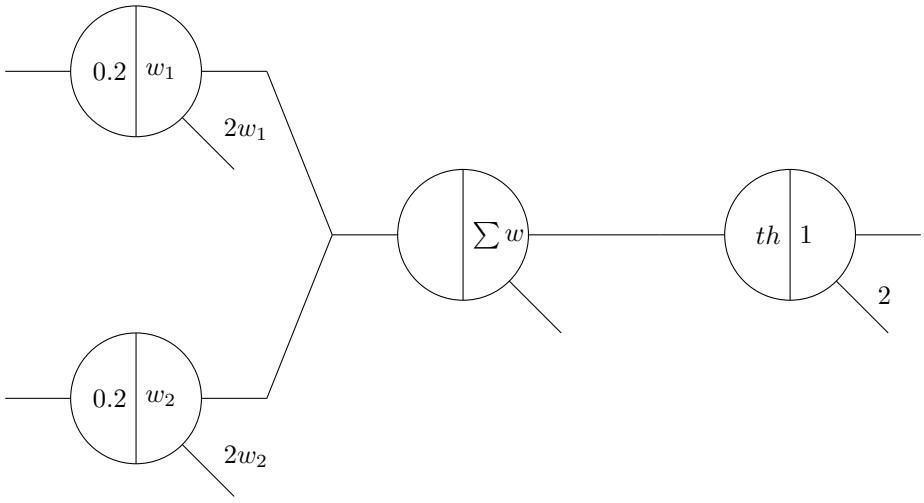


Figure 9: The seesaw neuron from Figure 7 shown schematically.

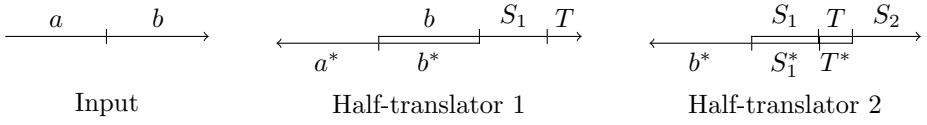


Figure 10: Translation of arbitrary input sequences into the syntax required for the seesaw neural network. The input strand ab displaces bS_1T from the first half-translator. bS_1T can then displace S_1TS_2 from the second half translator. This process successfully translates the input ab into S_1TS_2, which can then be used for input in a neural network (see Figure 7).

Bibliography

[1] Richard P Lippmann. An Introduction' to Computing with Neural Nets.

[2] Zhao Yanling, Deng Bimin, and Wang Zhanrong. Analysis and study of perceptron to solve XOR problem. In *The 2nd International Workshop on Autonomous Decentralized System, 2002.*, pages 168–173. IEEE Comput. Soc.

[3] Lulu Qian and Erik Winfree. A Simple DNA Gate Motif for Synthesizing Large-Scale Circuits.

[4] Lulu Qian, Erik Winfree, and Jehoshua Bruck. Neural network computation with DNA strand displacement cascades. *Nature*, 475, 2011.

[5] John M. Picuri, Brian M. Frezza, and M. Reza Ghadiri. Universal translators for nucleic acid diagnosis. *Journal of the American Chemical Society*, 131(26):9368–9377, 2009.