CS 443 / 525 Homework 2

Problem 1: XOR

Now that we can model single neurons, how about a neural network?

Let's say that we want to create a network that learns and can reproduce the XOR function:

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

1. How would you represent the inputs using a temporal encoding?
2. How would you represent the inputs using a firing rate encoding?
3. Build a 3-layer SNN using LIF or Izhikevich neurons. It should have two inputs (corresponding to the inputs x and y to the XOR function) and one output. The size of the "hidden" (middle) layer and the encoding of inputs and outputs are up to you. Each layer of neurons should be fully interconnected with the next layer:

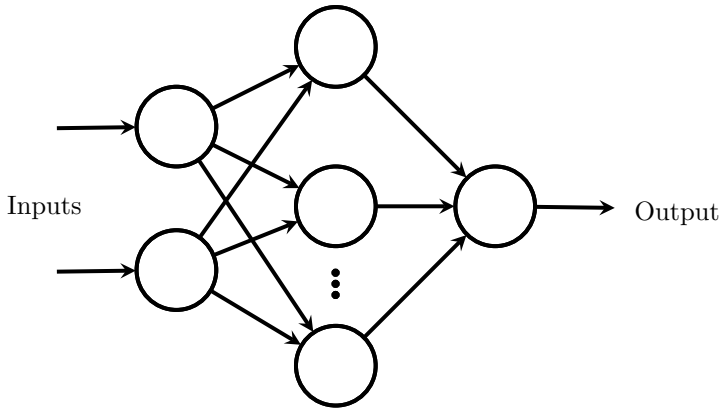

Figure 1: A 3-layer neural network with two inputs and one output

4. Train the network using Hebbian learning. Since this relies on correlations between neuron firing, you may need "teaching inputs" – additional neurons feeding into the last layer to force it to fire (or not fire) when it should (or shouldn't).

5. Train the network using STDP. You may need teaching inputs here as well.

Problem 2: Line detector

Retinal ganglion cells have so-called "center-surround" receptive fields that are excited by a stimulus in their center and inhibited by a stimulus in the surrounding area (or vice versa):

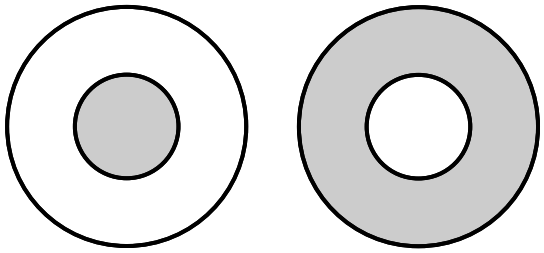

Figure 2: Receptive fields of retinal ganglion cells (on-off cells)

These cells are sometimes called "on-off" cells: they are either on-center, off-surround or off-center, on-surround. These cells cover the visual field, and their center-surround structure makes them very good at detecting edges in images.

Build a line detector network, using on-off cells, which responds selectively to lines with a certain orientation. The orientation of the line is up to you.

The network should consist of 2 layers. The first layer will be on-off cells that each take some portion of the visual field as input. The second layer will consist of the line-detector neuron.

Test its response for stimuli of orientations that span the 360 range, with 20 degree increments.

References

[1] Andrzej Kasiński and Filip Ponulak. Comparison of supervised learning methods for spike time coding in spiking neural networks. *International Journal of Applied Mathematics and Computer Science*, 16:101–113, 2006.

[2] Sen Song, Kenneth D Miller, and Larry F Abbott. Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature neuroscience*, 3(9):919–926, 2000.