**Genetics Artificial Intelligence**

We need a bunch of interconnected neurons. Some input and output neurons act as the connection to the outside world. The simulated world must be simple enough to take very little resources. There will be many brains at the same time, competing with each other for food, and for the rights to mate. Brains must have an impulse to eat to avoid dying, and to pass on their “genetic” material to their offspring. This genetic material dictates the starting connections of the neurons. Good genetics mean the brain is apt to improve itself over the course of its life. New neural connections can be sprouted spontaneously, or sprout randomly when neurons are fired. Upon being born, a brain will have neural connections randomly added to its brain, to allow faster evolution.

Connection to the outside world is accomplished via dedicated neurons. Every few ticks, the world is perceived through the input neurons, except for speech neurons, which are activated each time speech is heard.

Input neurons:

* 140 neurons for speech input (20 char)
* 8 neurons for position in 2D space (16x16 grid that loops around)
* 6 neurons for speaker recognition (max 64 IDs)
* 3 neurons for distance from speaker (max 8 diagonal cells)
* 1 neuron for hunger
* 1 neuron for food on cell
* 1 neuron for food nearby
* 1 neuron for pain (nearing death)

Output neurons:

* 140 neurons for speech output (20 char)
* 1 neuron for actually outputting speech
* 4 neurons for moving in the four possible directions
* 1 neuron for eating
* 1 neuron for breeding

Every time a speech output is made, every other brain is notified immediately.

A neuron fires when it’s signal level reaches a certain level. Every time a neuron fires, it adds to the signal level of connected neurons. How to avoid loops? By having neuronal connections to be directed [, and maybe weakening the signal strength when it propagates]. Let’s have each neuron accumulate signal until it fires.

Hyperparameters:

* ticks to wait between perceptions in input neurons
* weakening rate of neurons
* signal strength needed for neurons to fire
* decay rate for connections
* strengthening rate when firing
* [decay rate for signal propagation]
* Chance of new connection
* Chance of destroyed connection (based on current strength)

neuron states:

* 0 = ready
* 1 = just fired
* 2 = fired last tick

tick function

if tickCount == 0

for n in inputNeurons

n.level += getWorldInput

if n.level >= threshold

n.fire()

n.level = 0

else

n.level \*= weakeningRate

for n in neurons

if n.state == 0

if n.level >= threshold

n.fire()

n.state = 1

n.level = 0

else

n.level \*= weakeningRate

for n in neurons

if n.state == 0

n.level += n.toAdd

n.toAdd = 0

else

n.state = (n.state + 1) % 3

tickCount = (tickCount+1) % waitBetweenTick

fire function

for c in connections

if c.neuron.state == 0

c.neuron.toAdd += c.strength

c.strength \*= strengtheningRate

**Classes breakdown:**

* World: Simulates the outside world, a 16 by 16 2D world. Keeps track of the positions of each brain, and the size of the population. Generates food when appropriate.
* Brain: Contains many neurons, including some input and output neurons
* Neuron: Has a list of connections to other neurons. Has a current signal level indication when it's going to fire next.
* Connection: Points to a neuron, and has an associated strength value.