1. Energy Decay Network (EDeN)

Jamie Nicholas Shelley

Optishell consultancy

ABSTRACT

This paper and accompanying Python/C++ Framework is the product of the Authors perceived problems with narrow (Discrimination based) AI. (Artificial Intelligence) The Framework attempts to develop a genetic transfer of experience through potential structural expressions using a common regulation/exchange value ('energy') to create a model whereby neural architecture and all unit processes are co-dependently developed . The expression (Time and localisation of activation) are tuned stochastically by genetic experience; successful routes are analysed locally through application of tuned global rules: (Stability of signal propagation internal to cell (Process node)) (external state, internal immediate state, and genetic bias towards selection of previous expressions)).

These principles are aimed towards creating a diverse and robust network, hopefully reducing the need for transfer learning and computationally expensive translations as demand on compute increases.

Index Terms- Artificial, Energy, Entropy, Framework, General, Generative, Glial, Information, Intelligence, Model.

Contents

INTRODUCTION

Sections '*Genetics and genetic algorithms*' '*Nature of information and complexity*' and 'Artificial and biological neurons' are the authors observations and comparisons of neural computing from varying perspectives, this attempts to explain the reasoning behind the EdEN framework development. '*EdeN Framework and core process overview*' details application of this conjecture to a reduced cycle of operations designed to create a network of 'behavior driven intelligence'.

The section 'Artificial and Biological Neurons' details a neuron model ('Process node') that is evaluated by a common exchange value 'Stability index' which is assigned as a result of how well the node can manage energy locally over training (biased by product of historically successful influences ('Functome')).

This is designed to remove the need for direct data minimization using back propagation; replaced by training dynamic encoders that are flexible to varying levels of input (Or in extreme cases lack of).

Data input is synonymous with energy with a few key differences:

.Energy must be handled correctly by the network as to not cause excessive instability

.Energy input cannot be discarded by the system, it must be transformed to express network changes and behavioral outputs.

INTUITION

I. *The assumption that a neuron competes to survive in return of 'being a good signal processor' by which information can be dimensionally reduced and modeled. Mathematically this is the attempt to remove dependency on a global minimization function , replacing it with behavior that is translated to each unit differently depending on location and required processes of it's own 'survival', separate from training objectives.*

II. *The morphology and signal processing properties of the network are created from common principles/rules (as opposed to CNN architectures where architecture is manually defined in specialized layers) [Ref 11].*

III. *Genetics ('Functome') is expressed as a functional combination of an implication tree (L-system) that is genetically mutated where evaluated inputs locally express growth bias., ensuring a relationship between all development steps. This provides a mechanism for internally reasoned structural and functional definitions that are recorded for further potential intergenerational expression.*

IV. *As demand or dependency of a high entropy structure increases. As does a need for energy efficiency of it's operation. Once critical boundaries of this operation is met, structural representation of this process is maximized.*

A high level diagram of the developmental process:
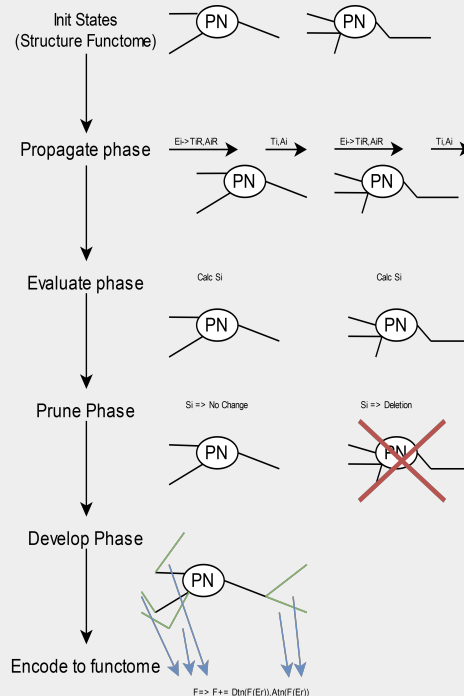
GENETICS AND GENETIC ALGORITHMS

In a typical GA (Genetic algorithm) **[Ref 10]** We create a base definition (gene) that is partially or completely randomised. A generation of genes are then tested against the desired outcome and mutated. (A specialized Monte Carlo method) Generations are merged by a percentage and manipulated against the results measured; leading to hopefully an exponentially appropriate solution.

Whilst this method with enough Compute/Time will eventually minimize, seemingly minor flaws in the loss functions, selection criteria often cause significant waste and fragility of the solution and result in increasing risk.

In the biological variant, expression of the gene is also encoded in the genome, with the crucial difference of encoded behavior/dependencies of the expression. This creates functional hierarchies that lead to further expression and regulation. As a result, biological genetics do not suffer from over specialization to the point of brittle collapse under environment change due to linked dependencies and regulation in every encoded item; even with far more complex encoded behaviors over generations, only stable extensions to the base rule set that correctly operate previously successful regulation are maintained.

Encoded information is expressed based on feedback through the existing environment (external and internal/(In contrast GA's typically train within a narrow scope).
Post expression, manifested objects (E.g Proteins) then operate within variance to also reinforce the environment expected of the genome, supporting further expression/regulation.

In contrast GA's are severely limited compared to the biological which comprises of structure, growth and execution definitions, not simply randomized/mutated words.
For a more details on standard genetic algorithms please refer to ref [14]

NATURE OF INFORMATION AND COMPLEXITY

I. Example in modern computing
The binary standard 8 bit byte. From which more abstract types such as float or long integers are constructed.
Base types interact through a common rule set (Logical (bitwise) or mathematical).
These processes are executed through registers which serve to perform ever higher abstractions through various languages.

All programming languages built on this architecture are interpretations and do not provide additional scope to the fundamental processes.

Notice that the bases of these types represent both base and structure, that is each bit of byte follows range for Br (Bit range) => 2^{Bi} (Bit index). This bit range is hierarchically dependent on the one preceding it.

II. How meaning is represented in compute
The order of precedence/use of the hierarchical building blocks defines how meaning is translated to human context.
This precedent is contractually arbitrary and then optimized through hardware.
For example, a hard-drive typically stores less frequently accessed but more critically dependent information than RAM. (even more so with an L1 Cache).
Processing of meaning requires highest entropy components, and storing requires lowest entropy components – HDD/SSD (Where structured information is most dense).

III. Biological Neuron comparison
In contrast, the brains most discrete transmission medium is an Ion.
Whilst groups of ions can hold a variable charge unlike a binary hierarchy, their function within neurons is binary -operating ion gates ; this discrete action operating over analog thresholds of ion concentrations resemble a compute architecture:
.Sodium and potassium ions to regulate charge as a response from direct electrical or neurotransmitter excitation (triggering an inbalance).

.Once a charge differential between external and internal environment is beyond a given threshold, the neuron fires to the axon terminals, reinforced/regulated by the Myelin sheath produced by Shwann Cells; continuing the potassium/sodium propagation to final Calcium inflow and neurotransmitter[s] release or direct electrical stimulation.

.The general Intensity of the stimuli is reflected in more frequent firings, where patterns of the stimuli reflect changes in the firing rate.

.In memory formation, groups of neurons grow to 'replay' memories of the past by generating the same collective output as before without the required chain of processed stimuli that created them -in other words an internal model, that is gradually more internally understood, (abstractedly similar to L1 to HDD process described in part 2 where clearly defined data structures are encoded).

.The frequency of firing reflects the neural coding of the stimuli. Various theories exist as to how this mapping encodes information precisely, however it is clear that the coding models the abstract information locally, with minor influences from global state; as apposed to back propagation in CNN systems: where error is translated to all layers. Please see refs 8, 9 and 12.

IV. Entropy and criticality
In both examples, points of high entropy correlate with least internally modeled information.
On the assumption that boundaries of hierarchical processes are defined by criticality of their operation. I propose the general rule applies in both nature and engineered computation: (Intuition section part IV:)
As demand or dependency of a high entropy structure increases. As does a need for energy efficiency of it's critical operation. Once critical boundaries of this operation is met, structural representation of this process is maximized.

Behavior driven minimization

Typically, data is trained with a defined objective within a coordinate space set. The results of training is then interpreted externally, as either an interface for the system or to assist in refinement of the continued training.
The assumption in this method of training is that data of some domain contains useful features, these features share common traits and can be binned. These bins are defined as a product of the requested minimization; training creates micro translations of features and their representations into the output.

Whilst this method works for simple solutions, the importance of the which features are best suited to minimize is quickly lost.

For example a successfully trained CNN is built to detect the difference between cat's and dogs.
All possible features of this complex domain are (assumed to be) within the model, however on testing a number of features happen to be contained within another, leading to less confidence or even the complete opposite output.
Whilst humans are also susceptible to this; a hierarchy of importance reduces this effect. If someone where to ask 'Do you like my cat' you are then biased visually towards looking for one, narrowing the scope of search criteria in an entirely different model before using the visual model, in other words : the application of a binned behavior is regulated by another to maintain validity.

Given this selection, the minimized output must be made aware to the more global 'selector' in order to determine the more appropriate response, this required relationship is similar to the architecture of a GAN (Ref 16); whereby discriminator informs the generator how close it is to the real data input.

GANs are however limited in the scope of the data as it is translated through each unit. EDeN attempts to solve this by bringing together the generator and descriptor into the same domain, more on the details of this in Ge/Di section under Eden Framework and core process overview.

Below depicts a high level control flow of this process:

Entity Neural Grid (3D space)

I/O transformation & Minimization

Internal self stimulus

Input Links

Output probes

Genetic expression from repeated stimuli

Genetic encoding of stable development

EDeN Server running on real Lidar data

EDeN Server running on Simulator (Unigine)

Text

Real time transfer learning

ARTIFICIAL AND BIOLOGICAL NEURONS

Artificial neurons work on the principle of a statically defined function/waveform that is then weighted at input set and singular output.
Results are evaluated through a loss function against the desired result from the end of the network; each neuron weight is adjusted to minimize the error against this output in relation to it's function [Layers examples: Ref 11] Neurons within hidden layers are minimized against the final output function relative to connected neurons. This encoding abstraction gradually trains a system to transform (and discriminate) data; applying back propagation through chains of partial derivatives.

Biological neurons attenuate their frequency of the coding pattern to predict the stimuli on input with no direct links to the required outcome [REF 7]
This implies there is not a uniform function to each neuron (As with Specialized Deep learning Layers). But base rules of how morphology, and intra/extra cellular events regulate to produce this function intrinsically from local and global environment.
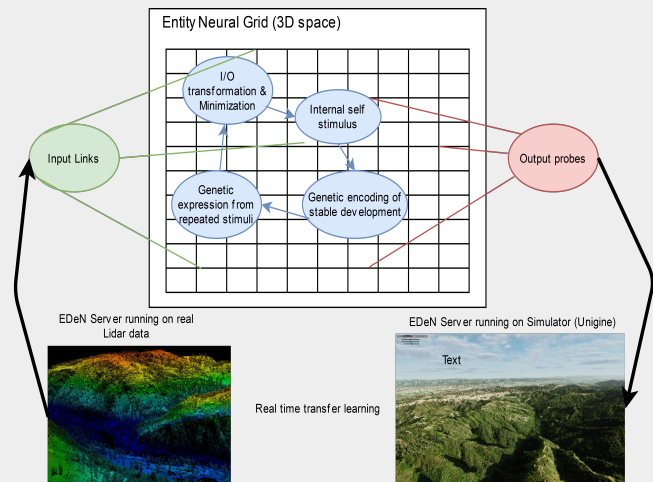
I. **Activation functions**
The Common Sigmoid function used in CNNs/Perceptrons operates in two dimensions and acts to exponentially decrease the effect of the weight summation beyond the mid-range values. In contrast to a biological neuron: both strength and frequency modulate **[Ref 12]** information. Neuro transmitter gradients, reflective dendrite/axon interactions (dynamic growth and pruning) provide many more options for specialization.

II. Propagation dependency
The 'Hodgkin Huxley' model, changes weights based on the error of the models output and the desired output.
This first makes all relationships inside the network strongly coupled to the information structure of the output, error correcting based on the value local to the network and global. In other words translating/discriminating input into output. (CNN Layers equate to a complex convolution filter).
It is unclear to what degree biological neurons are directly dependent on their surroundings, however the myriad of studied morphology/genetic dependent processes suggest a more resilient model than global 'moment guidance' methods used today. (SGD,Nesterov accelerated gradient, Adadelta/Grad etc [Ref 13])

III. Historical Neuron encoding
A Hodgkin Huxley and common spiking neural models don't encode the history of activation, they are updated iteratively to the immediate weight model.
In order to retain classification across multiple Input outputs, network models must therefore generalize results.

IV. Energy routing

The Eden Framework works to route 'energy' (a value that propagates over a Neural Grid) over multiple execution passes to build energy values internal to a neuron (process node), while other process node functions regulate this behavior. This allows for multi variate processing based on both external and internal state, (a kind of currency exchange). This medium provides a platform to apply all current advances in machine learning as well as experimental rules grounded in neuroscience.

EDEN FRAMEWORK AND CORE PROCESS OVERVIEW

I. The Neuron model:

Inspired by Self information theory. (Ref 6). Data is received from the training environment and inflicts instability on a the neuron model, this is recorded as a value to later apply morphological response, forcing an incremental improvement in the minimization and internal modeling of data. The neuron always produces an output as an attempt to integrate the input, by achieving this, safe regulation of 'energy' is maintained, minimizing the stability index to acceptable threshold values.

The morphology of the neuron model is represented as vector locations of the dendrites, and axon terminals, this produces delays and transformations in signal propagation by exchanging energy with transmitter indexes that have propagation

II. The Neuron model: Update Method (Process node IO):

On DevelopNetwork() function call , Existing neurons adjust their models as follows:

Ng = Neural Grid

E0 = current total energy value of the neuron

Et-1 = last total energy value of the neuron given spike (from soma, not necessarily output from an axon terminal)

Ei [E, v(XYZ)]= Energy at an input location To the Ngrid (Neural Grid); delivered by an input probe.

CEM = Currently Expressed Model (As a product of all neurites and soma process)

The CEM utilises the following components :

T[...]: An array of Axon Terminals – these modify the Process node output based on the propagation delay expected (Neural Grid Point Distance) from the source. On activation, they release a Transmitter index payload as a functome biased response from the EnergyValue.

D[…] An Array Dentrites – these provide regulation and modification to the Cem Fc metric and Cem's response .

Tt *At: Transmitter*/Architecture type, An Index and properties of the type if used by a Dentrite or Axon.

Variables unique to each specialisation determine the stimulation provided by the transmitter or morphology influence from architecture.

(This is designed to emulate the effect of ligand gates ION channel open/Closing on stimuli)

Gc[…]: An Array of Growth cones for each Dentrite or Axon that are regulated by the functome expression.

MaE Maximum total energy storage , a value above this will contribute to a decrease in the stability value of the Process node ; Indicating a lack of correct control. In combination with 'Average Energy Gated Response' values inside the CEM, value propagation is regulated into coded patterns.

MiE Minimum energy store before a spike, a consistent misfiring of the neuron due to forced response from extremes of energy will cause the stability of the neuron to decrease. (A tolerance of which causes pruning under phase control), this will result in random firings which do not filter out noise between correlated patterns observed by dendrite terminals.

Each Dentrite receives at least one TransmitterType, this type has a response that either blocks or allows energy updates to the process node.

'Architectors' are released based on measured acceptance criteria of a mutating functome to provide structured improvement to process node morphology.

Within the framework, transmitter and Architectors are combined to the type 'TransArch' Payloads.

### Ge/Di:

(Ge[neration] and Di[scrimination] states)

Generation occurs when the neuron model requires less input to stimulate encoded output that required to produce the original coding pattern; producing a predictive behavior.

This mechanism is also inspired by the prion theory of memory 'playback' by which only a fraction of the abstracted stimuli is required to reproduce the same signals.

Discrimination – the neuron receives more inputs than encoded output, (stability index is unchanged or decreases), the neuron is adjusting to new patterned stimuli and attempts to incorporate it into its existing Ge model. (If this is unsuccessful, stability index further decreases)

The Current Propagation Matrix is a result of the bias matrix and route matrix, that operate to produce output spikes when cross correlated data is enforced, and random is ignored.

A high level diagram of a process node expressing the axiom of a L-system definition and operating a single neuron
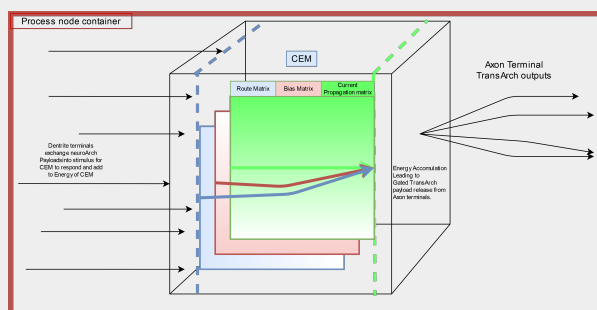
III. CEM processing

Given propagation, bias and router Tensors (P, B and R) of dimensions X*Y*Z, where X == Y and Z is N, N is user defined constraint.

The Energy value of a Cem where z = nth step across the tensors.

Below defines how PnE (Process node energy value) is determined.

It is a semi recursive algorithm across z of Z until AT(PnE) Axon terminal firing which regulates this value by TransArch exchange.

The Ro(Router options) is trained to apply a Bias and Energy value at P[x,y,z] to a given P[z]. Controlling different stimuli to reinforce (or weaken) E at P[z].

$$B_{xyz} = [0,1]$$

$$Ro = \begin{pmatrix} Ro_{11} & Ro_{21} & Ro_{31} \\ Ro_{21} & Ro_{22} & Ro_{23} \\ Ro_{31} & Ro_{32} & Ro_{33} \end{pmatrix}$$

$$R_{xyz} = \in Ro$$

$$F(P_z) = B_z \circ R_z$$

$$PnE(n) = \sum_{z=0}^{z=n} \left\{ At(PnE_{z-1}) + \sum_{x=0,y=0}^{x=X,y=Y} P_{xyz} \right\}$$

IV. Genetic model

Authors note:

The right abstraction to take in genetic representation took a long time due to at first wanting to create a language that produces generative functions similar to how a protein's 'process' is encoded (through Amino acids → RNA expression..) in DNA.

I decided against this due to the same behavior being plausibly expressed by morphology of the neuron and variation in signal model and vector based adjustment rather than computationally expensive micro instructions.

Rather than an incredibly complex genetic model of DNA transcription into protein folding and therefore function; the function of a select component in the network is already defined in terms of scope (Input, output, operation). Whereby the Functome either disables a suggested function of the L-system or biases the output.

For example:

Given a node has been suggested by an L system axiom and the Functome has agreed to the suggestion, a new process node will be added to the Neural grid.

.Input/output 'probes'.

As mentioned, to avoid a globally propagated minimization target that would enforce (excessive) bias over the processing units localized development, Inputs act as a standard vector update fields. Whereas output probes are vector readers that are used in training for monitoring or as part as physical entity simulation – with the intention of changing inputs in turn through, for example 'muscle' control (External to the Neural grid processes).

Inputs that are unexpected by the Neural Grids increase instability indexes across all neurons; forcing adaption or 'death' via phase analysis (discussed later).

V. *Initial growth structure and base growth*

The base network consists of predefined L-System with user defined functional variables.

The axiom of which represents process node growth and development.

The functome modifies this expression of the L-system by activating different functions based on process node related feedback.

A 'ChangeLog' component of the neuralContainer records all changes from initial creation to adjustments made per growth Phase execution. This is later compared against the Fuctome in order to add updates.

A new Entity that contains an updated functome as a result of ChangeLog. Future generations of the Functome expressed by the L-System will then be able to advance more quickly given a similar environment as well as contribute to it's own solutions during development.

This improves both the regulation of axiom expression and the structure of the network in relation to the overall ability of the network to handle unpredictable inputs.

An analogy to this methodology is to compare human and ape's language ability: Humans clearly have a genetic bias to the general architecture required of speech, which is then specialized at approximately the same age.

A simple example:

A sub component in the initial rule could be:

VPz+1 → (Pn-1E>(0.2)) * Fn[CLId](E))

Vpz = vector point z component

Pn-1 – process node nth -1 (Previous process node in the z domain)

E = energy

FN[CLId] = functome function at Change log Id (Indexor) with an energy input of 0.2

(Translating to: Add 1+Functome bias to vector component Z if the previous process node energy level is above Functome 0.2, with the Functome bias also receiving this value). The Functome function over generations then has the ability to turn on/off this growth or regulate against ever more complex dependencies as attached at the index CLId (analogous to a codon in genetics).

Note: Originally, energy functions contained a 'leak' decay value inspired by the uncertainty of neuron efficiency and a proven useful tool in deep learning, I have since removed this following ref 17; a study where a third family of ion gated channels account for this leak – in other words all Functome and l-system energy updates must be conserved.

Architecture affects and 'Progenitor' regulation

Similar to neurotransmitter, 'Architectors' (An analogy to neuropeptides) exist to create new Process nodes and other static influences on the neural grid given specific prerequisites that apply to all other Neurites. *(Frequency and Density of Transmitter Indexes that provide discrete energy updates to the internal state of the unit)*. Both Initial Axioms and Functome definitions provide details as to what Process nodes are produced in terms of their activated. That is the assumption of the requirement to a select Neurite configuration before the environment argues for it's existence against normal Growth cone calculations. In contrast to Deep Learning or other static network definitions. This provides an element of 'disposability' to each Process node.

That is, a given neurite configuration could be created with the intention of being entirely unstable, never fulling integrating with the Entity architecture; instead being used as a bias to other more static Process Nodes.

A requirement of this development exists in reference 13, whereby higher abstraction in the visual domain correlated less with direct stimuli, however it's unknown how this pattern projects to other existing biological domains or if what the boundary conditions of dimension returns are.

Non L-system functions

Recognizing the need for extra cellular regulation that is modeled from the three types of glial cells: the Functome doesn't exclusively operate on the L-system domain. Other Functome functions are used during phase execution; during the phase executions of the entire network, the maintenance of previously 'grown' process nodes are controlled as briefly described below.

Example of this need is in Agent propagation:

Typically GA's do not control agent propagation; rather they provide the researcher with results for external selection criteria of the experiment, however for correct adjustment of the function and future L-system propagation, selection criteria must be (albeit highly abstracted) embedded in the Functome.

Functome overview

Unlike a genome, the Functome's complexity of initial axioms is reduced; 'Primordial soup' did not contain a predefined hierarchy of objects (although one can argue it's base axioms are based on restrains in physical rules). Nonetheless, the Functome acts to encode behavior during all stages of entity development without explicitly enforcing expression.

Each type and sub type contains a Functome reference, a lookup range to the area's of the Functome where all non-volotile data is recorded; that is all fundamental states, of which without; the object would be pointless, for example, the position on the Neural grid.

The Functome's secondary purpose to bias the expression of algorithms inside each construct in relation to the initial axiom data.

With these factors combined, using types 'ExpressionRequirement' that is created based on the change log - a map of how process nodes are architected over the entity lifespan is defined.

Execution Phases

Changes to the network are updates via 4 main steps.

Output probes and input links propagate energy updates via transmitter exchange to each entity.

For example, adding the value 0.5 to an input every second (without Functome bias), with a network update duration of 0.5 seconds will leave the input probe value at 0 before each update period as there a complete exchange of value.

Note: this method is designed to encourage self regulation of input and leads to internal contextual differences in data being equally reflected in unique sub architectures within the expressed entity.

Propagate network:

The L system re-write rule executes against the Functome rules of expression (Each type), using inputs and current internal energy state to calculate all energy updates, energy is exchanged for 'transmitters' and 'architectors' at specific locations across the neural grid.**Evaluate Network:**

The Stability index is calculated for all process nodes and all global readers are updated, this includes actuators for example which would move an agent in order receive input probe updates.

This is calculated by:

Prune Network:

Using the stability index against non-L system Functome functions. Process nodes and all neurites that have 'failed' or reached their planned life span (emulating controlled cell death) are removed.

Develop Network:

The Process nodes are allowed to be modified based on Evaluate Network results where all vector, new process node and neurite updates are executed. The L system of linked functions that continue to propagate within this nodes are further refined.

With architecture options encoded into the functome, the selective expression is then driven by immediate processes in the Neural grid .

Workflow description and future Roadmap

Depicted left, a high level diagram of how data is propagated between variations of the EdeN Server and EdeN Clients.

Given successful entity training. The Functome or simply the saved state of the entity is saved.

A copy of this save state can either be 'locked' - whereby only the propagate phase is activated or used for further development under potentially different environments or within a simulation/embedded context with other entities with alternative specialized.

This system encourages transfer learning as a core principle of development, rather than an afterthought to training.
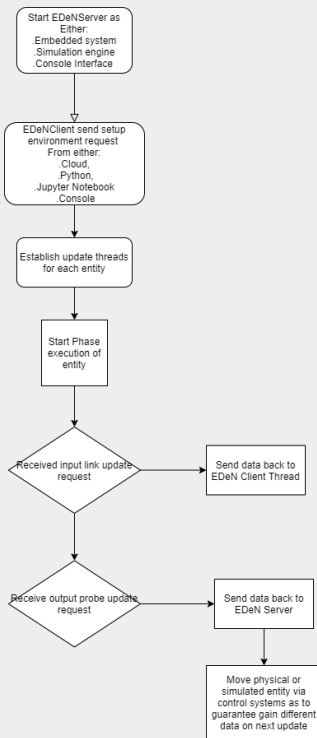
Minimal example

TODO

REFERENCES

[1] **Thermodynamics for a network of neurons : Signatures of criticality.**
Gaˇsper Tkaˇcik , Thierry Mora, Olivier Marre , Dario Amodei, Michael J. Berry II, William Bialek
 (https://arxiv.org/pdf/1407.5946.pdf)
22-Jul-2014

[2] **Towards Cytoskeleton Computers. A proposal.**
Andrew Adamatzky , Jack Tuszynski , J¨org Pieper, Dan V. Nicolau, Rossalia Rinalndi , Georgios Sirakoulis, Victor Erokhin, J¨org Schnauß,David M. Smith
(https://arxiv.org/pdf/1810.04981.pdf)
11 Oct 2016

[3] **Glia - Neuron Interactions in Neurological Diseases: Testing Non-cell Autonomy in a Dish**
Kathrin Meyer, Brian K. Kaspar.
(Research Institute at Nationwide Children's Hospital, Columbus, Ohio, USA 2Molecular, Cellular & Developmental Biology Graduate Program, The Ohio State University, Columbus, Ohio, USA 3Department of Neuroscience, The Ohio State University, Columbus, Ohio, USA)
(https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4939136/pdf/nihms765160.pdf)
11 Feb 2018

[4] **A Mathematical Theory of Communication**
C.E Shannon
(http://math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf)
Jul/Oct 1948

[5] **Glial cells in neuronal network function**
Alfonso Araque* and Marta Navarrete
(https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2894949/pdf/rstb20090313.pdf OR)
12 Aug 2010 -Aprox

[6] **Neural code – Neural self-information Theory on How Cell-Assembly Code Rises from Spike Time and Neuronal variability**
Meng Li 1 and Joe Z. Tsien
(https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5582596/pdf/fncel-11-00236.pdf)
30, Aug, 2017

[7] **Pyramidal Neuron as Two-Layer Neural Network**
Panayiota Poirazi*, Terrence Brannon and Bartlett W. Mel
(https://www.sciencedirect.com/science/article/pii/S0896627303001491?via%3Dihub) March 2003

[8] **Automatic fitting of spiking neuron models to electrophysiological** recordings
https://www.frontiersin.org/articles/10.3389/neuro.11.002.2010/full
05 March 2010

[9] **Maximum Entropy Approaches to Living Neural Networks**
Fang-Chin Yeh , Aonan Tang,, Jon P. Hobbs,Pawel Hottowy, Wladyslaw Dabrowski , Alexander Sher, Alan Litke, John M. Beggs
(http://www.beggslab.com/uploads/1/0/1/7/101719922/19yehetal2010.pdf)
2010

[10] **Learning with Genetic Algorithms: An Overview**
Kenneth De Jong
(https://link.springer.com/content/pdf/10.1007%2FBF00113894.pdf)
15 Jan 1998

[11] **List of Deep Learning Layers**
(https://www.mathworks.com/help/deeplearning/ug/list-of-deep-learning-layers.html)
Revision 2019

[12] **Single Neurons may encode simultaneous stimuli by switching between activity patterns**
Valeria C. Caruso , Jeff T. Mohl, Christopher Glynn , Jungah Lee ,Shawn M. Willett , Azeem Zaman, Akinori F. Ebihara , Rolando Estrada, Winrich A. Freiwald,Surya T. Tokdar, & Jennifer M. Groh1
(https://www.nature.com/articles/s41467-018-05121-8.pdf)
2018

[13] **A large-scale, standardized physiological survey reveals higher order coding throughout the mouse visual cortex**
Saskia E. J. de Vries, Jerome Lecoq, Michael A. Buice, Peter A. Groblewski, Gabriel K. Ocker, Michael Oliver, David Feng, Nicholas Cain, Peter Ledochowitsch, Daniel Millman, Kate Roll, Marina Garrett, Tom Keenan, Leonard Kuan, Stefan Mihalas, Shawn Olsen, Carol Thompson, Wayne Wakeman, Jack Waters, Derric Williams, Chris Barber, Nathan Berbesque, Brandon Blanchard, Nicholas Bowles, Shiella Caldejon, Linzy Casal, Andrew Cho, Sissy Cross, Chinh Dang, Tim Dolbeare, Melise Edwards, John Galbraith, Nathalie Gaudreault, Fiona Griffin, Perry Hargrave, Robert Howard, Lawrence Huang, Sean Jewell, Nika Keller, Ulf Knoblich, Josh Larkin, Rachael Larsen, Chris Lau, Eric Lee, Felix Lee, Arielle Leon, Lu Li, Fuhui Long, Jennifer Luviano, Kyla Mace, Thuyanh Nguyen, Jed Perkins, Miranda Robertson, Sam Seid, Eric Shea-Brown, Jianghong Shi, Nathan Sjoquist, Cliff Slaughterbeck, David Sullivan, Ryan Valenza, Casey White, Ali Williford, Daniela Witten, Jun Zhuang, Hongkui Zeng, Colin Farrell, Lydia Ng, Amy Bernard, John W. Phillips, R. Clay Reid, Christof Koch
(https://www.biorxiv.org/content/10.1101/359513v1.full)

[14] **Optimizing gradient descent**
(http://ruder.io/optimizing-gradient-descent/)
19 January 2016

[15] ***On the practical usage of genetic algorithms in ecology and evolution***
Steven Hamblin
(https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.12000)

[16] ***Dynamic routing between Capsules***
Geoffry E. Hinton
(https://arxiv.org/pdf/1710.09829.pdf)

[17] ***Generative Adversarial nets***
Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair , Aaron Courville, Yoshua Bengio
(https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf)

[18] ***The Sodium "Leak" Has Finally Been Plugged***
*Terrance P. Snutch1, * and Arnaud Monteil2 1Michael Smith Laboratories, University of British Columbia, Vancouver, BC, Canada V6T 1Z4 2 Institut de Ge´ nomique Fonctionnelle, CNRS UMR5203 - INSERM U661 - Universite´ s Montpellier I et II, De´ partement de Physiologie, 34094 Montpellier Cedex 5, France*
(https://www.cell.com/action/showPdf?pii=S0896-6273%2807%2900339-X)

[19] Robinson C, Hou L, Malkin K, Soobitsky R, Czawlytko J, Dilkina B, Jojic N. Large Scale High-Resolution Land Cover Mapping with Multi-Resolution Data. Proceedings of the 2019 Conference on Computer Vision and Pattern Recognition (CVPR 2019). (https://www.microsoft.com/en-us/ai/ai-for-earth)

Authors

Jamie Nicholas Shelley, Optishell Consultancy

Journal revision reference here