Version 2.01 Last updated: 11/04/2021

Energy Decay Network (EDeN)

Jamie Nicholas Shelley
Optishell consultancy

DOI: 10.31224/osf.io/dfyzn

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 by genetic and real time signal processing influences; successful routes are defined by stability of the spike distribution per epoch which is influenced by genetically encoded morphological development biases.
These principles are aimed towards creating a diverse and robust network that is capable of adapting to general tasks by training within a simulation designed for transfer learning to other mediums at scale.

Index Terms**-** Artificial, Energy, Entropy, Framework, General, Generative, 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 (influenced by product of historically successful morphology changes that are genetically encoded (‘Functome’)).

INTUITION

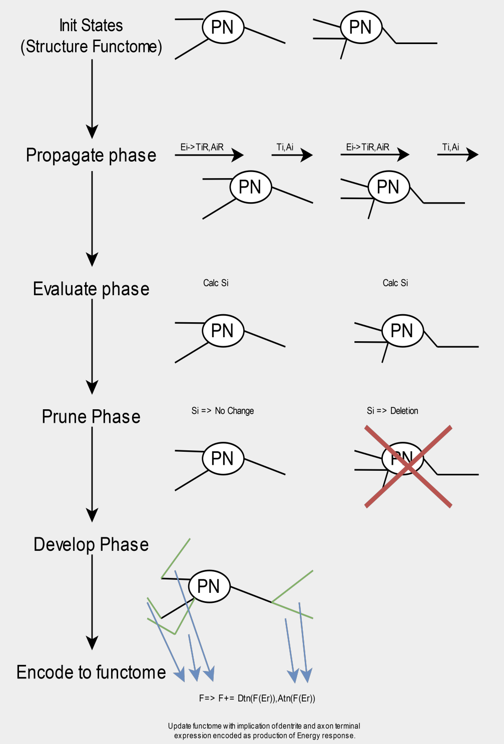
I. The assumption that a neuron (process node) competes to survive in return of ‘being a good signal processor’ by which information can be dimensionally reduced and modeled. Mathematically this is the attempt to remove dependency on a global minimization function , replacing it with behavior that is translated to each unit differently depending on location and required processes of it’s own ‘survival’, separate from global entity 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 morphological biases from internal environmental evaluations; ensuring a relationship between all development steps. This provides a mechanism for internally reasoned structural and functional definitions that are recorded for further cross domain utilisation and intergenerational expression.

IV. As 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 (bit-wise) 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 in-balance).
.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. **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:)

Behavior driven minimization

In deep learning, 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 through a network, training for global minimization leads to information loss that would otherwise be useful in assisting specific inferences.

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.

By training for network activity when some degree of domain related information is present, micro features can be trained and combined to provide a flexible and more reliable inference.

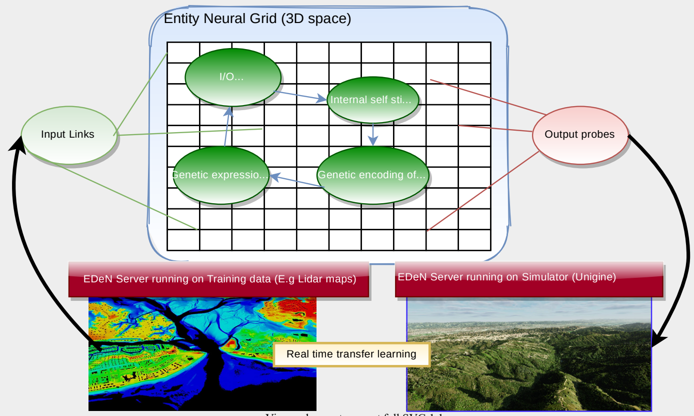
For example if someone where to ask ‘Do you like my cat’ you are then biased visually towards looking for one, narrowing the scope of search criteria in an entirely different domain before utilising the visual model, in other words; the application of a binned behavior is regulated by another.

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.

In contrast, an EDeN process node specialises to one or more stimuli by requiring less influence on the input dendrite terminals to produce the same spike for a given pattern (More on this later).

Below depicts a high level control flow of this process:



Cross correlation of supportive data from multiple domains at different frequencies, combined with local reinforcement of non random signals per process node (or neuron) then removes the requirement for specific global minimization goals.

In summary the pressure of the entities survival over time (more on stability index later) trains for specialisation per domain required of it.

Training the entity for a given task is to provide a reacquiring set of inputs interlaced with expected noise/variance.
The effect of an output probe on the input must then change the action to input to repeat itself within minimal variance in order for the entity to specialize.

Internally this forces both a model of the input/output relationship and a means to ignore irrelevant patterns.

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 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.

.Activation functions
The common functions 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. Neurotransmitter gradients, dendrite/axon interactions (dynamic growth and pruning) provide many more options for specialization.

. Global 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. **[Ref 13].**

.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 generalise results.

.Energy routing
The EDeN Framework works to route 'energy' (a value that propagates over via Architectors and exchanged to the CEM , influencing energy propagation). Multiple execution passes 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).

FRAMEWORK AND CORE PROCESS OVERVIEW

. 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, stimulating morphological response, leading to an incremental improvement in the minimization and internal modeling of data.
The neuron produces an output after building a minimal energy value via internal weights and routers. These act to minimize a 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 'transmitters' or 'architectors' that are analogous to neurotransmitters and neuropeptides.

. The Neuron model: Update Method (Process node IO):
On DevelopNetwork() function call, Process Nodes adjust their models using the following variables

Ng = Neural Grid
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)

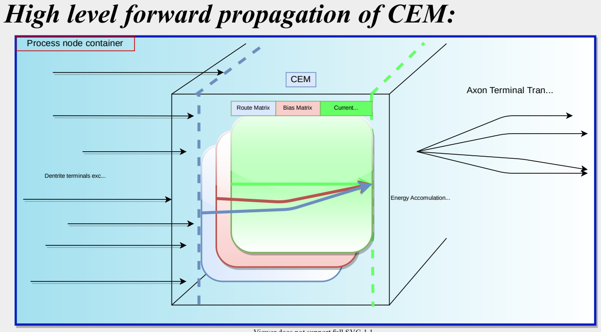
The CEM utilises the following components:
T[…]: An array of Axon Terminals – these modify the Process node output based on the propagation delay expected (Neural Grid Point Distance) from the source.
On activation, they release a Transmitter index payload as a Functome biased response from the EnergyValue.
D[…] An Array of Dentrites – these provide regulation to the CEM influence
Tt/*At*: *Transmitter/*Architector type, An Index and properties of the type if used by a Dentrite or Axon. (defined as TransArch payloads in code)
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) and influence LTP like behaviour (long term potentiation).**Gc**[…]: An Array of Growth cones for each Dentrite or Axon that are regulated by the Functome expression.
MiE Minimum energy store before a spike, a consistent misfiring (or suppression) 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.
-Referenced as 'TransArch' Payloads in the framework.

PWR 3D Tensors, (Propagation, Weight and Router)
Work to produce spikes across each Z axis value, training for spike response from input sets per process node (See more information under CEM processing section)

Generation/Discrimination states:
Generation occurs when the Process Node model requires less input stimulus to produce a previously encoded response than that required to produce the original pattern. *(This process begins to occur post spike when the route and related weight matrix has been correctly enforced).*
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 Process Node receives more inputs over an epoch than it was previously trained for, (stability index is unchanged or decreases), the Process Node is adjusting to new patterned stimuli and attempts to incorporate it into its existing model.

High level forward propagation of CEM:



Viewer does not support full SVG 1.1

.Current Energy Model (CEM) processing
The CEM acts to tune different stimuli to reinforce (or weaken) E at P[z].Given the propagation, weight and router Tensors (**PWR**) of dimensions X*Y*Z, where X == Y and Z is N, (N is user defined constraint), energy value E of a CEM where n is z index of an XY across the tensors.
Depicted right defines how PnE (Process node energy value) is determined.
A recursive algorithm across z of Z until PnE of the z plane is above the minimum threshold.
This triggers axon terminal firing which regulates this value by TransArch exchange to the external Neural grid, at which point all propagation values are reset to 0,
The Axon terminal fires at delayed time by decrementing against the initial distance from the Neural container and the axon terminal location.
Note: Other regulating mechanism may be applied in the future, as influence by homeostatic regulation in biology ***[ref 20]***

$$RO_x = \begin{matrix} \square & Ro_{010} & Ro_{020} \\ \square & Ro_{011} & Ro_{021} \\ \square & Ro_{012} & Ro_{022} \end{matrix}$$

$$RO_y = \begin{matrix} \square & Ro_{110} & Ro_{120} \\ \square & \square & Ro_{121} \\ \square & Ro_{112} & Ro_{122} \end{matrix}$$

$$RO_z = \begin{matrix} \square & Ro_{210} & Ro_{220} \\ \square & Ro_{211} & Ro_{221} \\ \square & Ro_{212} & Ro_{222} \end{matrix}$$

$$R_{xyz} \in RO$$

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

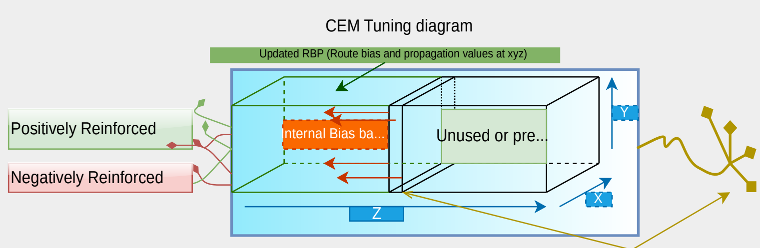
$$R_{xyz} \in \{0,1\}$$
$$W_{xyz} = [0,1]$$
$$PE_{xyz} = W_{xyz} R_{xyz}$$
$$PNSD = \{x > 0 \mid x \in \mathbb{R}\}$$
$$DR = \{x > 0 \mid x \in \mathbb{R}\}$$

$$Assuming \sum_{i=0}^{|PNSD|} \left(PNSD - 1_i - PNSD_i\right) \geq 0$$

$$PN_{SI} = \frac{1}{max\left(\left(KL\left(PN_{SD-1} | PN_{SD}\right)\right), 1\right)} - DR$$

$$R_{xyn-1} = f\left(PE_n\right) = \begin{Bmatrix} E(RO), (PE_n < MinE) \\ Ro_{121}, (PE_n \geq MinE) \end{Bmatrix}$$

Each router value (See R0 for router options) variable at n-1 xy plane on spike is updated to directly route to the spiking n plane if spike propagation energy (PE) is above Minimum Energy (MinE). Otherwise the value is set to a random variable of the



Viewer does not support full SVG 1.1

Router options (E(R0)).

Updating the Stability index of the process node (PNSI) involves calculating the KL divergence across the previous Spike distribution (SD) and current SD per epoch, minus the decay rate (DR) if the process node is unresponsive.
This has the effect of pruning non responsive process nodes given input. Once a spike occurs during an epoch, the biases for the propagation reinforcements are tuned by standard **SGD** (Stochastic gradient descent) (The target vector being dynamically created on spike) starting from the spiking z position, reinforcing (negative or positive) the dendrites.

This has the effect of reinforcing the particular router configuration to spike at the same z position in the future, but requiring less input from the dendrite terminals.
Meanwhile any other inputs that are received in different phases are propagated through first the previously reinforced router/weight configuration up to z, then beyond (As they may not reach the minimum energy threshold).

.Input/Output probes.

To avoid a globally propagated minimization target that would enforce 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 or monitoring 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 have repetition then increase the stability indexes across all neurons; forcing adaption or 'death'.

. Genetic model and Functome overview

Authors note:
The right abstraction to take in genetic representation was first motivated by how a protein's 'process' is encoded (through Amino acids → RNA expression..) into 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.

The Functome acts to encode behavior during all stages of entity development as options for future re-expression.
Each type and sub type within the framework contains a Functome reference. Examples include positions for the initial structure of an entity and action options.
Generations of Functome expression lead to increased adaptability of the process node architecture under different input conditions.

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 affinity to the general architecture required of speech, which is then specialised at approximately the same age.

.Functome Encoded actions

As the develop network phase is activated, 'AvailableActions' per Process node are scanned for by checking a prerequisite name which selectively utilities any additional 'value' and 'direction' data.
The mutation update modifies a collection of these commands per process node, references of this data is then stored in the Functome.

Action names:
.*AxonTerminal_Add*
.AxonTerminal_Remove
.Dentrite_Add
.Dentrite_Remove
.DirectConnection_Add
.DirectConnection_Remove
.AddTransmitterIndexCreator
.AddArchitectorIndexCreator
.AddTransmitterIndexResponse
.NeuroGenesis
.Apoptosis

Prerequisite names:
.*TransmitterIndexPresent*
.Density
.AllowOn_EntityInternalClockMin
.DisableOn_EntityInternalClockMax
.ArchitectorIndexPresent
.*Energy*

. Initial growth structure and base growth

We start with random available actions created per NeuralContainer, with random positions for cell body, axons and dendrite terminals encoded to the Functome.
On each epoch, neural containers execute these growth patterns, achieving changing stability index over time. Actions continue to be evaluated and mutated during the develop phase of the execution. This in combination with following existing growth rules slowly adapts the morphology of the process node.

Architectors

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 actions. That is the assumption of the requirement to a given 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.

A requirement of this development exists in ref [13], whereby higher abstraction in the visual domain correlates 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.

Execution Phases Per Epoch

Propagate network

Output probes and input links propagate energy updates via transmitter exchange to each entity (see section ‘The Neuron model:’ above for more details).
The CEM internally propagates it’s energy updates and set’s spike data if it occurs. This Spike data (Z index of the spike and total XY Plane energy) is used to tune the routers, update the current spike distribution, which is then used to back propagate the CEM weights for specialisation.
‘TransArchPayloads’ are then exchanged from CEM onto the neural grid based on the parent process nodes Functome encoded instructions.

Evaluate and Prune Network

The Stability index is calculated for all process nodes, those that fall below the
minimal threshold are removed – emulating neural pruning.

Develop Network

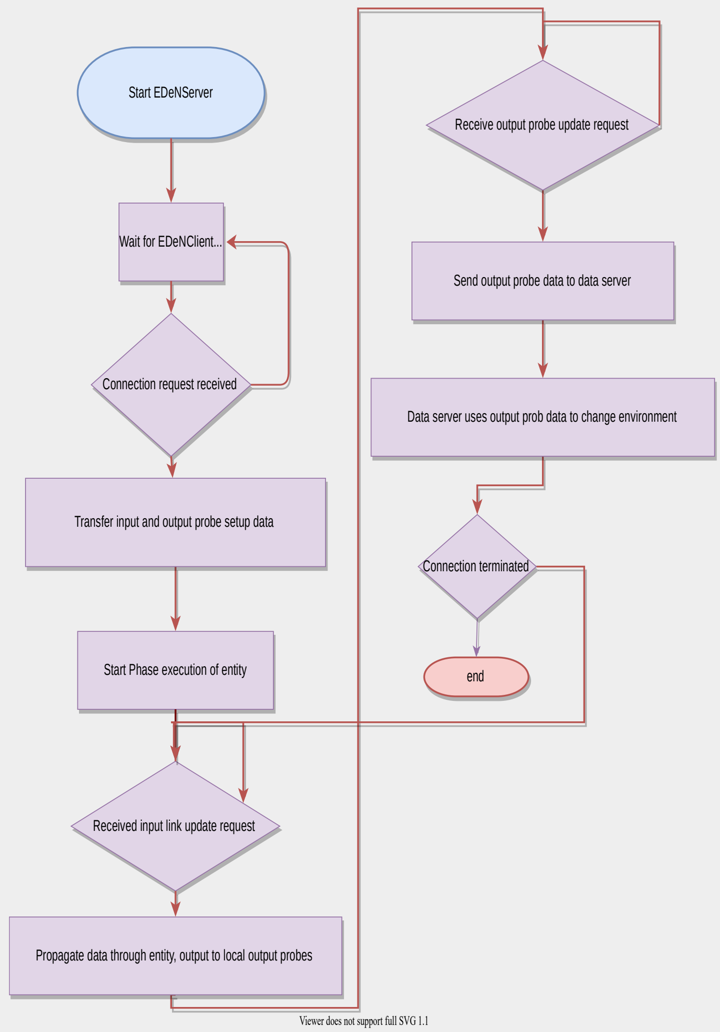
The remaining Process nodes are allowed to be modified based on Evaluate Network results, new process nodes and Neurite updates are executed. The active Functome functions continue to propagate within the process nodes.
Mutations are applied randomly to the growth rules/actions if the stability index is below a set minimum.

Workflow description

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

Given successful entity training. The Functome mutation is disabled.
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.



Closing comments and next steps

Whilst all mechanisms described above function as expected within the framework, large scale testing for trained entities performing complex tasks has yet to be achieved, further research and development work continued in this area and more papers are to follow with results.

It is believed the methods selected in this framework will provide a better interface to general training without requiring specialist knowledge, leading to the authors desired goal of supportive/ duplicate systems to support sectors from healthcare to R&D and manufacturing.

References and Influential sources

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
12 Aug 2010~

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

20 ***The Self-Tuning Neuron: Synaptic Scaling of Excitatory Synapses***
Gina G. Turrigiano
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2834419
31 Oct 2008

Authors

Jamie Nicholas Shelley, Optishell Consultancy