Evolutionary Computing and Artistic Expression

Daniel Famiyeh

Research

Definitions

Neural Networks (NNs) are systems modelled on the human brain and are 'trained' to give a certain output based on a set of inputs. Inputs to a neural network are 'weighted', meaning that they have differing levels of influence on the output.
Inputs with higher weights, have a bigger impact and different inputs and weights result in vastly different outputs being produced. The inputs and outputs are interconnected through layers of 'neurons' which produce different outputs based on inputs and their respective weights. When a neural network is being trained, it is given a set of inputs and compares the output it calculated against what the output should have been. From this, it calculates its' error. Based on this error the network makes tiny 'nudges' to the weights of its inputs to push it in the right direction of getting the right output. This process is repeated with different inputs until enough nudges are made to the weights of the network that it gives the correct output based on a certain input. At this point the network is said to be 'trained'.

Genetic Algorithms (GAs) are algorithms that are inspired by natural selection whereby 'genes', which can take the form of any arbitrary data type, are passed down via a (chromosonal) *crossover function.* These genes have a chance to mutate and are often scored with *fitness function* and it is this score that is used to aid selection.

Neuroevolution is a method of generating the topology and parameters of a NN using genetic algorithms. The topology of the network describes the way in which the neurons are arranged and how they are connected. The parameters of the NN entails the values of the weights. So simply neuroevolution is a method of generating a neural network that is functionally fit by applying rules of natural selection.

A *TWEANN* is a ***T****opology and* ***W****eight* ***E****volving* ***A****rtificial* ***N****eural* ***N****etwork* and as the name implies, it is a NN that evolves its structure as well as the weighting of nodes.

1) The Bibites

The Bibites is a virtual ecosystem made with Unity in C#.
All behaviour is evolved with respect to a set of inputs and outputs woven together to form neural networks that evolve over time via a genetic algorithm known as NEAT.

Inputs

State
Constant
Hungriness
Maturity
%Health
Speed

Vision
Distance from closest Bibite seen
Angle to closest Bibite seen
Distance to closest pellet seen
Angle to closest pellet seen
Number of Bibites seen
Number of pellets seen
Bibite colour R
Bibite colour G
Bibite colour G

Clock
Clock - tik-tok, on-off, Hz-¿genes
Clock Chronometer
Clock - time alive

Pheromones
Pheromone Sensing 1
Pheromone Sensing 2
Pheromone Sensing 3

Outputs

Movement
Forward
Backward
Left
Right

Wants
Want to lay
Want to eat
Want to mate
Want to grab

Biological
Reset Chronometer
Pheromone Sensing 1
Pheromone Sensing 2
Pheromone Sensing

Initially the Bibites do nothing, but there is a chance for mutations which means that they may move in a certain direction.
By doing this they may eat a pellet and lay eggs passing on their genes.
Overtime through natural selection they evolve to:

- Steer towards food.
- Pick up and save food if they're full.
- Develop memory.

The Bibites have genes that are expressed in phenotypes such as:

- Size
- Speed
- colour
- Mutation Factors

Cost Functions

- $Metabolism\ Cost = \frac{SpeedRatio}{2 \times SizeRatio}$

Maintaining Equilibrium

A result of evolving the Bibites' brains is that they become too good at survival. To solve this, initially, the amount of pellets spawned was limited in such a way that it is inversely proportional to the number of Bibites alive. Having max 200 Bibites alive at any given time is the aim.
Let P be the number of pellets and B the number of Bibites, then:

$$P = 100 * \frac{200}{B}$$

However, this only stabilises growth and does not necessitate that the equilibrium be at 200. As a result of this, the pellet spawn equation must also be scaled down with a constant as long as there are more than 200 Bibites alive.
This is given by:

$$P = K \times \frac{200}{B}, \quad \text{where } K = K \times \exp(\frac{200 - B}{200 * C})$$

where C is an arbitrary tweaking parameter.
After improvements were added to the ecosystem, spawining pellets was a function of a new 'energy model'.
Bibites eating habits exist on spectrum with carnivorous existing on one side and herbivorous existing on the other. More carnivorous Bibites will tend to fight other Bibites to eat them and upon dying Bibites drop meat as well as having remains turn into fertile waste to spawn new pellets. This system of predation ensures that the ecosystem limits population growth more autonomously.

The next two sections of research are based work by K.O Stanley and R. Miikkulainen written in their research paper 'Evolving Neural Networks through Augmenting Topologies.'

2) Shortcomings of generic TWEANNs; Introducing NEAT

NEAT stands for ***N**eural **E**volution (through) **A**ugmenting **T**opologies* and is a genetic algorithm technique in neuroevolution tailored towards natural selection by evolving the topology of neural nets as well as their weights.
In order for net to be able to evolve the topology will have to be encoded in order to form a genome that can be used for crossover.
The two primary encoding schemes for TWEANNs are:

- *Direct* - Every connection and node that will appear in the phenotype is specified in the genome.
- *Indirect* - Rules are specified for generating a phenotype; Either layer specification or growth rules. Allows for more compact representation than direct.

Binary Encoding (Direct)

Traditional GA bit string representation used by Dasgupta and McGregor in their *Structured Genetic Algorithm* method (1992). In their implementation a bit string represented the connection matrix. The algorithm is known for its simplicity in operating like a standard GA. There are some limitations with this methodology:

1. Matrix size is directly proportional to square of nodes
2. The max number of nodes must be chosen by a human since bit strings must be the same size for all organisms.
3. Using a linear representation of graph makes it difficult to ascertain if crossover will yield successful combinations.

Graph Encoding (Direct)

Splits representation into grid and linear sub-components. The grid sub-component represents the graph and the linear genome provides node definitions and specifies connections. The graph representation is used for sub-graph crossover and topological mutations while the linear component is used for point crossover and parameter mutations. This implementation was used by Pujol and Poli in 1997 in their *Parallel Distributed Genetic Programming* (PDGP) system.
Limitations:

1. Limit on max number of nodes based on grid representation.
2. Combining certain subgraphs might not produce functional offspring.

Cellular Encoding (Indirect)

Genomes are written in a graph transformation language that specify cell division which in turn result in different types of connectivity. Genes are compact and can be used multiple times in network development by requesting cell division at a different location. CE can encode the development of a network from a signle cell.

NEAT uses direct encoding.

Competing Conventions

The *Competing Conventions Problem* is one of the main problems that arises from Neuroevolution.
It arises from the fact that for a network with n hidden neurons there are $n!$ functionally equivlent solutions. This results in potentially important import evolutionary information being lost during crossver which can lead to a less functionally sutiable network being produced. This is particularly a problem in TWEANNs due to a lack of constraint on the topologies that can be created. NEAT aims to address this by addressing the problem of representing different structures as this is the root of the problem. Representations may not necessarily match up. Genomes may be of different sizes, genes themselves may appear at the same point at an indivdual chromosome but express a completely different phenotypes, while genes expressing the same phenotype may appear at different positions on differing chromosomes. This is an issue that is also paralleled in nature wit respect to gene alignment in sexual reproduction as genomes, as they appear in nature, are not of a fixed-size through *gene amplification* (Darnell and Doolittle, 1986; Watson et al. 1987); The process by which new genes are added to a genome. These new genes can't just insert themselves anywhere into a genome without some sort of 'awareness' of which gene is which. This problem is solved in nature via *homology*; Two genes are categorised as homologous if they are alleles of the same trait. Homology as a concept, is a difficult one to implement in neural networks so NEAT uses a *historical origin* concept between two genes; Two genes are defined as homolohous if they are both derived from the same origin. Thus a process of *artificl synapsis* (the process of lining homologous genes) is achievable allowing for new structures to be created while keeping track of the types of genes themselves and the phenotypes they represent.

Speciation

Adding new structures to a net will cause the fitness of a network to decresae intiially as the weigEachht of the new node has not had time to optimise. Because of the initial loss of fitness, the ikelihood that a new structure will survive in a popoulation long enough to optimise are low. Naturally, there needs to be a way to preserve innovational structures long enough for them to be optimised. In nature, new structures are implicitly proteced as different structures specialise in different niches. Thus isolating network structures in to niches woud give them a chance to become useful before competing with the general popultaion.I In NEAT, speciation is achieved with *explicit fitness sharing* which means that agents of similar genomes share their *fitness payoff* (Goldberg and Richardson, 1987). In NEAT, networks are grouped on their topology, weight parameters and genetic history; limiting the number of networks that can exist on single fitness peak. The population is divided into species on different fitness peaks without any single species taking over. So innovations ar protected on a per-species basis.

Topological Diversity and Innovation

Random initial topological populations ensure topological diversity from the outset but produce problems in TWEANNs. There is a chance that nets will be produced that have no path from inputs to outputs which take time to filter out of the population. Random initial populations to do not result in finding minimal solutions in which the number of parameters to be searched is reduced, since from the beginning, NNs in the population have extraneous nodes ad connections. Minimising such NNs means getting rid of these components that should not have existed to begin with rather than recombining topologies to achieve minimistaion. Larger networks will dominate as long as they have a high fitness as there is not function to check them against. One method to solve this is to penalise a network bsed on its size. A problem with this is what do we define to be network large enough to deserve penalisation? Even if an answer for that can be agreed upon how much do we penalise said netowrk by? Solutions to different problems necessitates, different structures so its hard to know whether or not any defined fitness function is, for sure, pushing neuroevolution to a minimesd solution. NEAT solves this by starting out with a minimal configuration, meaning no hiden nodes, and structures only continue to grow if growth benefits the solution. This ensures that the system traverses the lowest-dimensional weight space for all geneartions, minimising the final product and all intermediate problems.

Whats so neat about NEAT?

The Algorithm

NEAT centres on two types of structural mutation; adding nodes and adding connections between nodes. Each gene as a corresponding innovation number that maps the ancestor of each gene, new genes are assigned higher numbers. Every time a new connection is made, a new *connection gene* is added to the end of the genome and is designated another innovation number. When a new node is added, the connection gene that has been split is disabled and the two new connection genes are added to the end of the genome. A node gene is also appended to the genome.

Genetic Encoding

Genomes are linear representations of network connectivity containing *connnection genes* and *node genes*. The former describes two nodes being connected while the latter lists inputs, outputs and hidden nodes that can be connected. Connection genes specify the in-node, out-node, connnection weighting, whether the gene is expressed and an innovation number. In structural mutation, the new connection to the new node has a weighting of 1 and the connection from the output of this node to the node previously connected has the same weighting as before. This minimises the initial effect of mutation and means that although a slight functional dissimilarity can arise due to the introduction of non-linearity in the change of connection, new nodes can be quickly integrated into a netwrork with ample time to optimise.

Gene Tracking and Alignment

As previously stated, additions to the genome are given a *unviersal innovation number* to keep track of when they were introduced into the topology. When genomes mate, offspring will inherit the same innovation numbers n each gene. There is a chance a piece of structural innovation can receive differeing innovation numbers in a single generation if it occures more than once via single-generational mutations. However this can be attenuated by giving each identical mutation the same innovation number. During crossover *matching genes* with the same innovation number are lined up while *disjoint* and *excess* genes are not. Genes from etither parent are chosen randomly at matching genes, while excess and disjoin genes are always passed from a 'fitter' parent, albeit randomly.

Speciation with NEAT

The number of excess and disjoint genes between two genomes are indicative of difference between two topologies as species. A greater disparity suggests a greater *compatibility distance*, δ and thus a likelihood that the two genomes originate from sparesely differeing genetic histories. Therefore δ is a linear combination of excess and disjoint genes, E and D respectively as well as the average of weight differences of their matching genes (including disabled genes) $\bar{W}$:

$$\delta = \frac{e_1 E}{N} + \frac{e_2 D}{N} + e_3 + \bar{W}$$

The e coefficients allow for control over the importance of the 3 factors and N is the number of genes in the larger genome, so that the excess and disjoint gene factors are normalised. N can be set to 1 for small genome sizes. Using this we can then speciate with a compatibility threshold δ_t and thus an ordered list of species can be maintained. Existing species are represented by a random genome of species of a previous generation so that species don't overlap. If a genome, g, is not compatible with any existing species a new species is made with g of as its repesentatitve. Explicit fitness sharing necessitates that organisms of the same species share the fitness of their niche. The adjusted fitness of an organism is given by,

$$f_i' = \frac{f_i}{\sum_{j=1}^{n} sh(\delta(i,j))}$$

The function $sh = 0$ when $\delta(i,j) > \delta_t$ otherwise $sh(\delta(i,j)) = 1$. Therefore the sharing function is limited to agents within the same species. Reproduction is then achieved by first eliminating species that perform the lowest with respect to their niche and not to the population as a whole, preserving topological innovation.

Minimising Dimensionality

This achieved implicitly via NEAT by starting out with a uniform population without hidden nodes. New structures are introduced via structural mutations and only those structures deemed useful through fitness evaluations survive. Thus the search space is minimised as NEAT is always searching through a minised dimensional space with growth that converges towards a solution.

4) The Art of Artificial Evolution - A Handbook on Evolutionary Art and Music (J. Romero and P. Machado)

Basic Principles

- The design problem must be presented to the computer numerically
- By specifying traits of a design we make a *parametric model* that implicitl defines a *solution space*
- These parameters are our *genotype* and the design created is the *phenotype*
- A *population* of designs is created by varying the values in our genotype
- In evolutionary art its more common to use a mathematical expression as the genotype

When using an expression as the genotype we arrange it to take the form of a tree structure; With functions and operators at nodes and constants or variables at the leaves. Systems like these are often called examples of genetic programming.

Artificial Art Made by Artificial Ants (N. Monmarché, I. Mahnich, and M. Slimane)

Ants roam a 2D plane i.e. a bitmap image and paint as they traverse the image. They move stochasitcally and thus tend to preserve direction. We define three parameters to P_1, P_2 and P_3 as the probability that the ant will turn left, right or go straight ahead. Naturally, $P_1 + P_2 + P_3 = 1$. The paint placed on the canvas is a colour specific to each ant and is indicative of said ants' pheromones defined by RGB components (C_R, C_G, C_B). At each step, the ant also looks for the pheromone path of another ant, the colour of this path defined by components (F_R, F_G, F_B). A luminance comparison is performed:

$$Lum(R, G, B) = 0.2426.R + 0.7152.G + 0.0722.B$$
$$\delta(F_R, F_G, F_B, R, G, B) = |Lum(F_R, F_G, F_B) - Lum(R, G, B)|$$

If δ is less than a give threshold the ant recognises the colour of the path and follows it with a probability of P_s. For diffusion, the colour laid via a *discrete convolution product* on the neighbouring cells. This is performed by the matrix:

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

From Swarm Art toward Ecosystem Art

Research paper by S. Bornhofen, V. Gardaux, A. Machizaud
Swarm Art-Creating a system of individualised agents who work together in producing generative art. The agent has a:

- **Location** - Position of agent.
- **Orientation** - Direction agent is heading.
- **Speed** - Distance covered per step.
- **Curvature** - Rate of curvature in movement.
- **Energy** - Positive real number which is used up in movement. If it falls below 0, the agent 'dies'. This can be replnished by picking up items.
- **covColor** - A colour which changes based on the colour of items ingested. The agent is a mix between this and their genotypic colour.

and genotypically posses these qualities:

- **Irrationality** - Degree of variation in the curvature.
- **Fecundity** - probability of producing a child at each step.
- **Offset** - Positive value for the offset of a child's angle from that of its parents.
- **divRatio** - Energy an agent allocates to the child it produces.
- **sensorRange** - Range of perception.
- **Energy Consumption** - per unit step in time.
- **Agility** - The capacity to orient to a target.

- **prColour** - True colour.
- **prStrength** - Strength of true colour.
- **linSize. expSize** - Linear and exponential parameters for size of agent at a given energy level.

Reproduction

Asexually modelled and children posses exact copy of parent genotype. A reproducing agent passes state variables: *location*, *speed*, *curvature* and *covColor*. Initial energy and orientation are defined by parental *divRatio* and *offset* gene.

Food Chasing

When food spawns in an agents perception range of an agent the will turn towards it with repsect to their agility gene. Food give the agent a certain amount of energy and contain an intrinsic colour. Colour and size change with respect to a set of rules.

Colouration and Size Equations

$$\begin{aligned} size &= linSize * energy^{expSize} \\ covColor &= (covColor * energy + fColor * fEnergy)/(energy + fEnergy) \\ color &= (prColor * prStrength + covColor * energy)/(prStrength + energy) \end{aligned}$$

The Plan

The organisms will individual pixels or small polygons who will posses a:

- **Position Vector** - for the location of the agent in space.
- **Velocity Vector** - for the speed and direction the agent is heading.
- **Energy** - Positive real number that is consumed in movement and reproduction. The alpha component of the agents RGBa value will will be directly proportional to this.
- **Colour** - Mix between colours consumed and true colour.

Genes that will be inherited include:

- **Energy Consumption** - Energy consumed per time step.
- **Curvature** - Curvature of path.
- **Max Speed** - Max speed of movement possible in lifetime.
- **Agility** - Ability to turn towards targets such as food.
- **Range of Perception** - Radius with which the agent will be able to perceive the world.
- **True Colour** - Phenotypical colour inherited from parent
- **Initial Energy** - A portion of parent's energy.
- **Initial Velocity** - The initial velocity will be a unit vector in the direction of the parent.
- **Mutation Rate** - The probability that any single gene should have a mutation, this too can mutate.

Reproduction

Any agent will have a unviersal arbitrary reproduction rate that will only come into effect after 10% of it's live.