EVOLUTIONARY COMPUTING

NEUROEVOLUTION

BY DR. KARINE MIRAS

Quick survey + Q&A

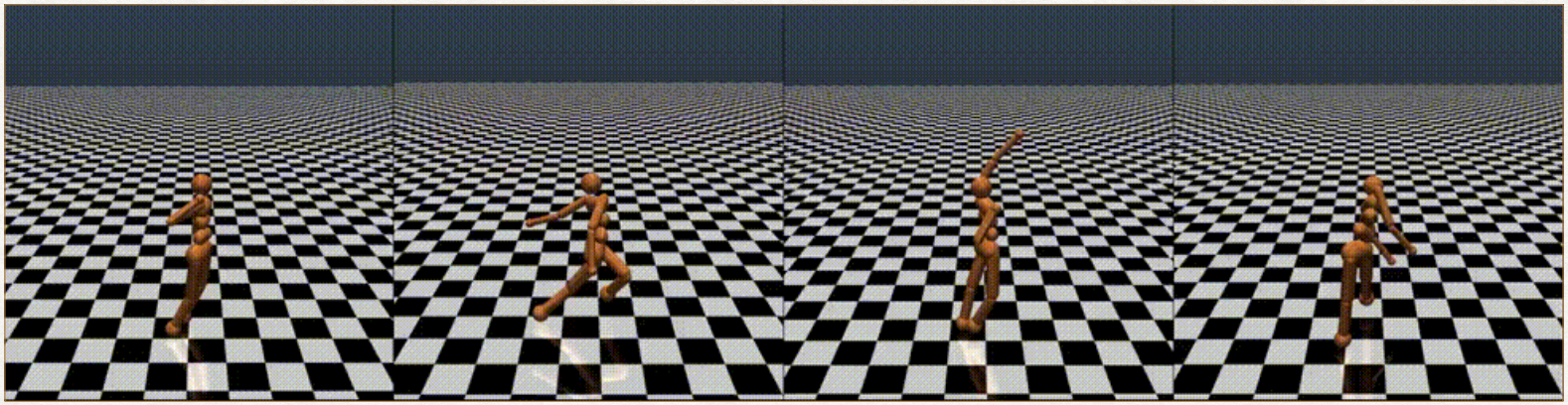
The code **3866 4238**

Mentimeter

NEUROEVOLUTION

Applications: virtual creatures

Evolving locomotion with Evolutionary Strategies

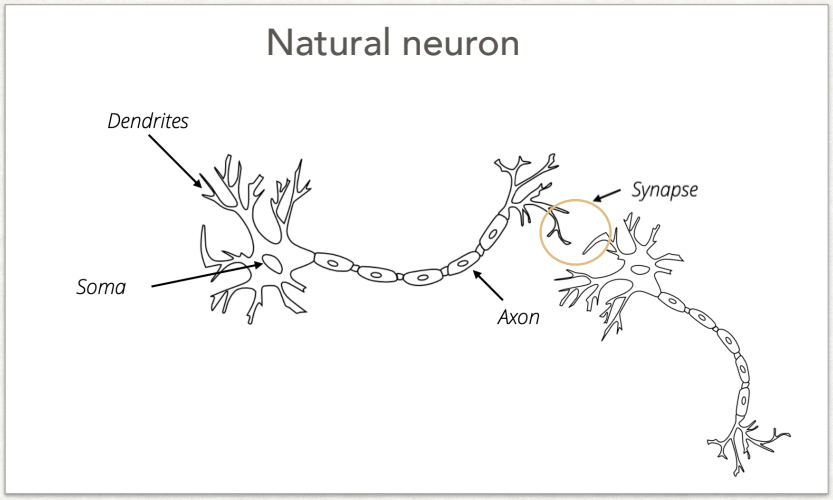


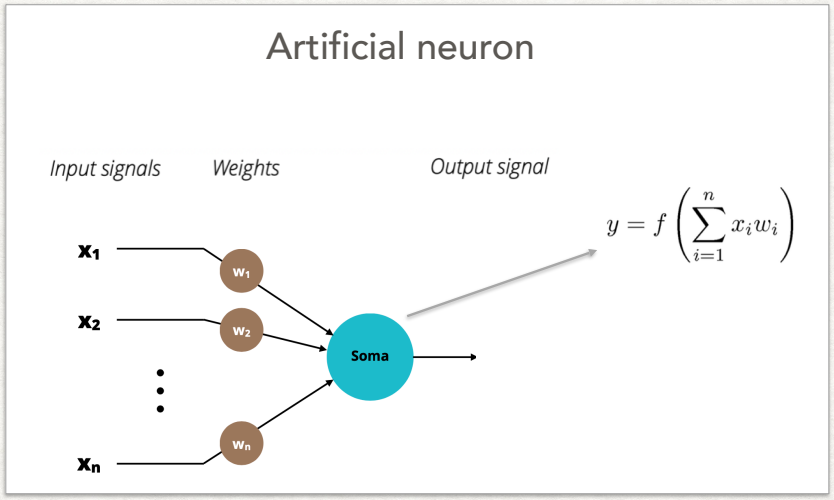
https://blog.openai.com/evolution-strategies/

ARTIFICIAL NEURAL NETWORKS

Intro

A very simple abstraction of a natural brain.

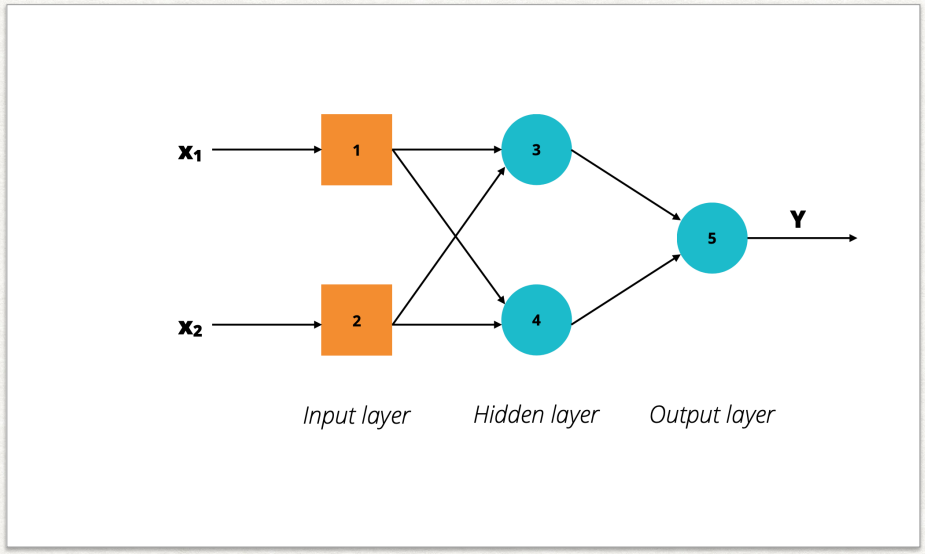


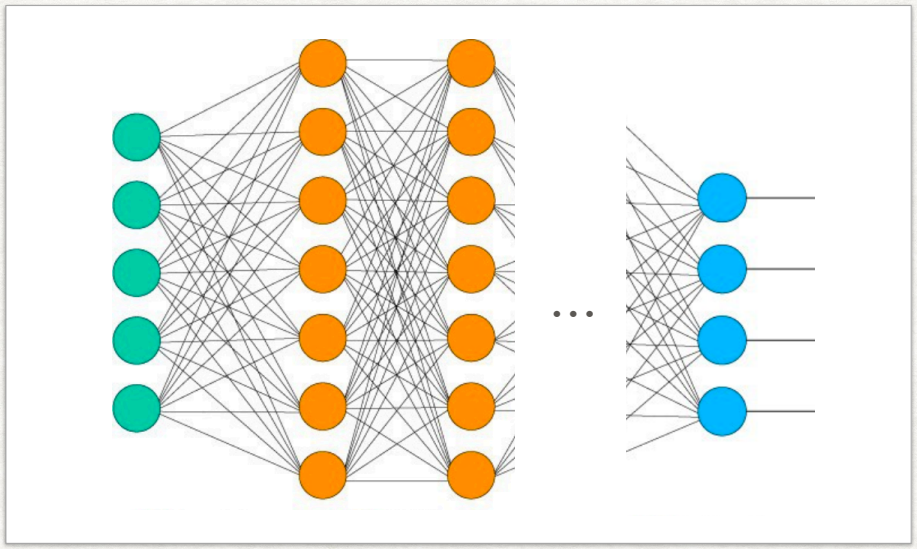


ARTIFICIAL NEURAL NETWORKS

Intro

Multiple layers



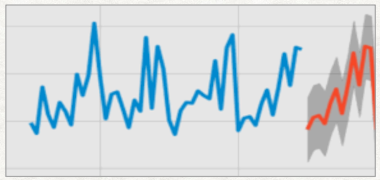


ARTIFICIAL NEURAL NETWORKS

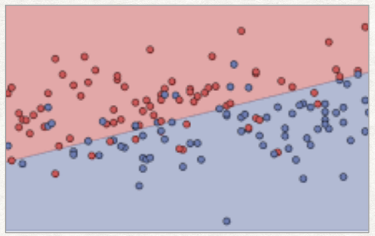
Common applications

Prediction and classification

- How much are we gonna sell next month?



- Will this client payback a loan?



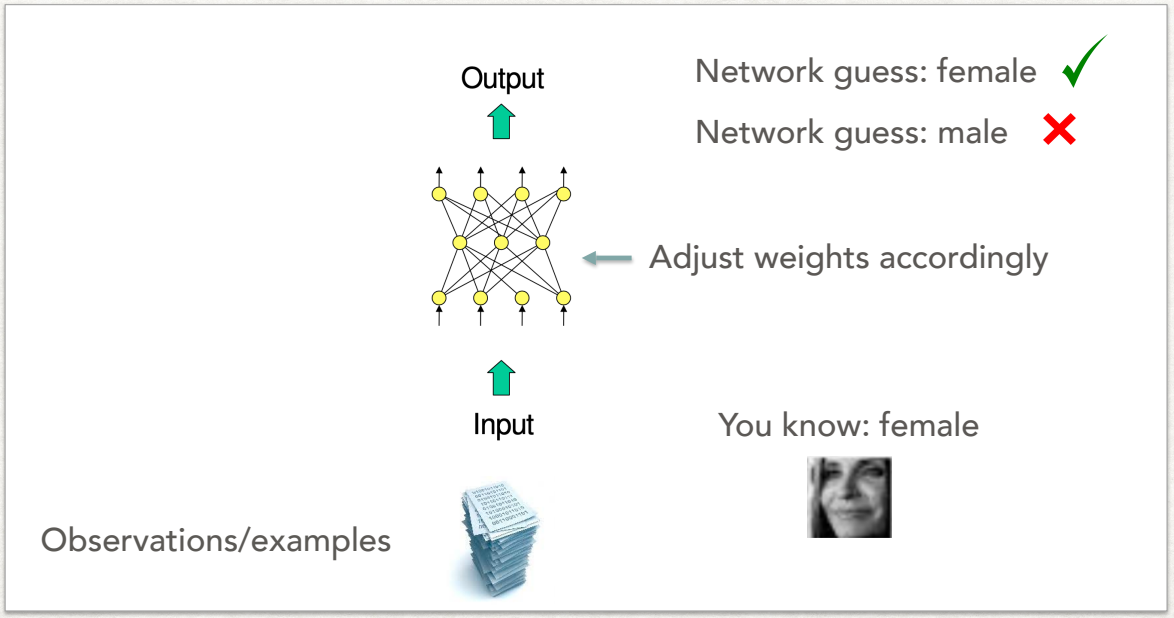
- What is the hair colour in the images?

ARTIFICIAL NEURAL NETWORKS

Common learning method: supervised learning

You know the answers!

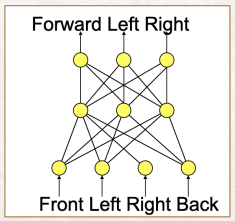
Algorithms, example: Backpropagation



NEUROEVOLUTION

What if you do **not** know the answers?

Example of application: control



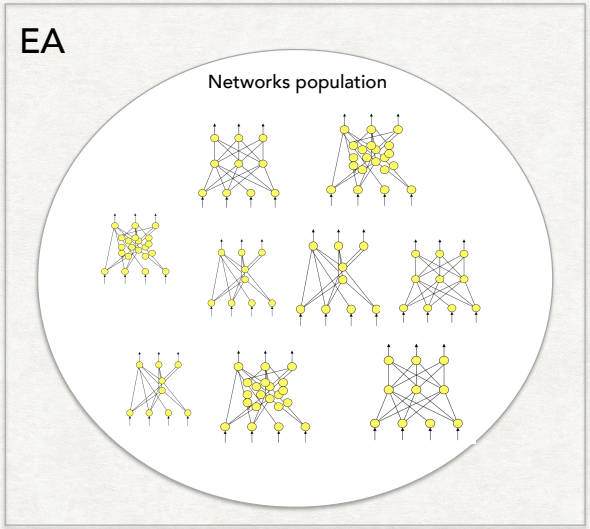
Unfortunately: we do not always have a set of samples.

How do you know what is right?

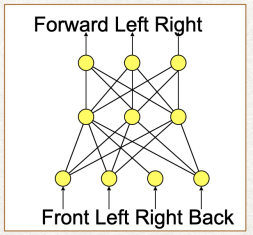
There are many ways of driving "well"…

NEUROEVOLUTION

Instead of adjusting weights of one network through supervision of examples, evolves populations of networks.



Fitness: performance of the network on given task



NEUROEVOLUTION

Main variants:

> evolving only the **weights** of a network with fixed topology

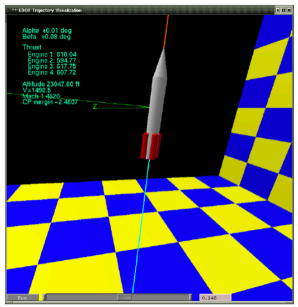
> evolving also the **topology** of the network

> evolving also the activation **functions** of the network

NEUROEVOLUTION

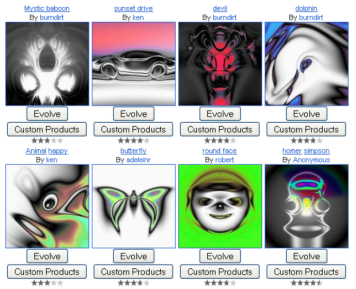
Applications

Often used for **control** tasks and **generative** challenges.



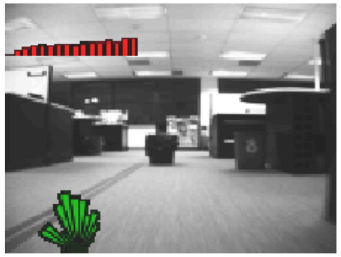
Rocket Control

Video Game NPC Control

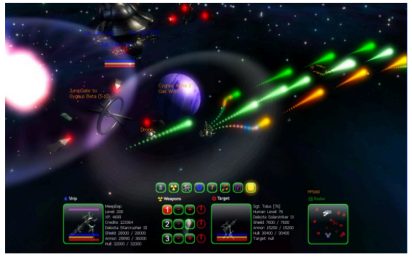


Evolving Pictures

Evolving Music



Real-world Robot Control



Video Game Content Generation

…Prediction and classification problems are also possible!

NEUROEVOLUTION

Pros and cons

- Can be used for supervised, unsupervised, and semi-supervised learning tasks.
- Can avoid local optima by explicit diversity maintenance methods (because EAs are population based).
- Combining multiple criteria as objective
- Creative and interesting solutions, instead of just "perfect solutions".

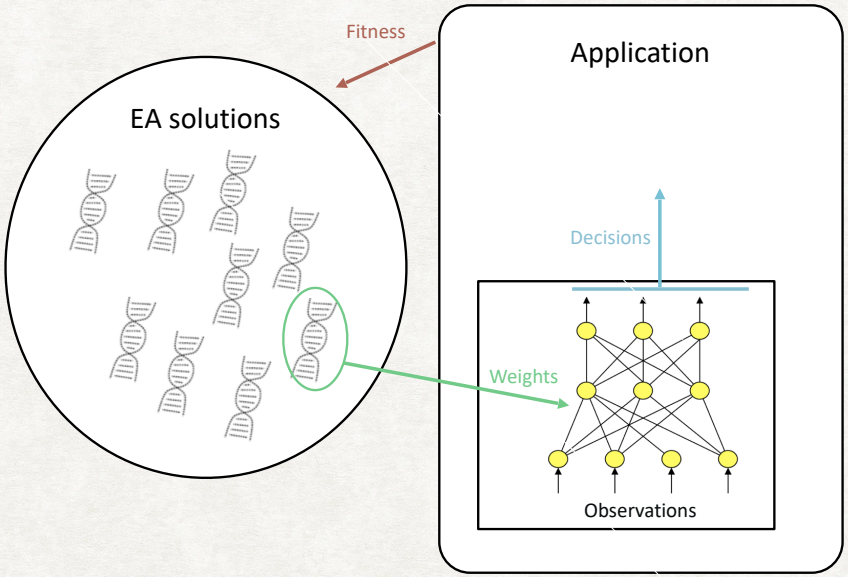
- Computationally expensive in comparison to learning methods (a whole population of solutions to be tested).
- What else…?

SIMPLE NEUROEVOLUTION

Evolving only the weights

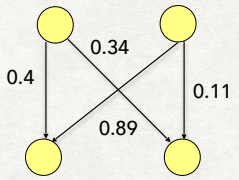
- Transforms NN learning problem into vector optimization.

- Fixed topology, often fully connected.



Solution 1: 0.1, 0.3, 0.9, 0.6
Solution 2: 0.4, 0.34, 0.89, 0.11
…
Solution N: 0.23, 0.9, 0.4, 0.12

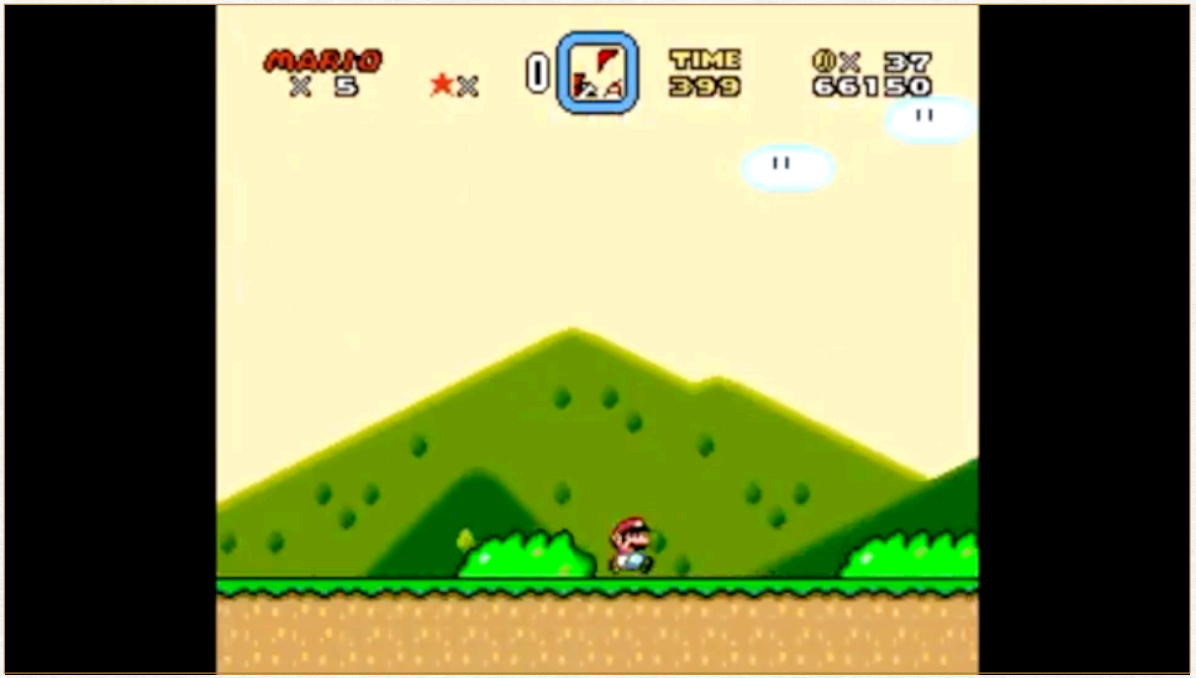
Example:



NEUROEVOLUTION

Applications: games

A neural network with a genetic algorithm to play Super Mario World: evolving only weights.

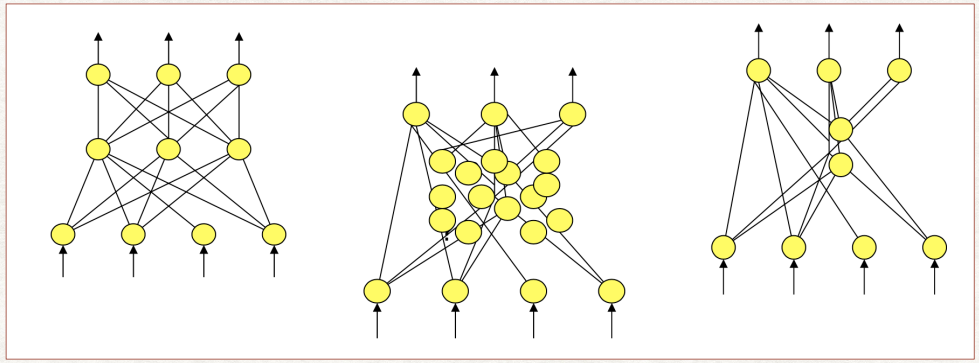


https://www.youtube.com/watch?v=qv6UVOQ0F44

ADVANCED METHODS

What if a fixed "standard" topology is not good for the problem at hand?

What is the best topology?



- Make the topology also evolvable:
  Topology and Weight Evolving Artificial Neural Network (TWEANN)

- And even the activation functions

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Evolving weights and topology

Main premises:

> **minimality**: topology complexity as minimal as possible

Networks bigger than necessary are **expensive** to optimise!

> **protecting of innovation through speciation**

Give innovations a chance to show their potential

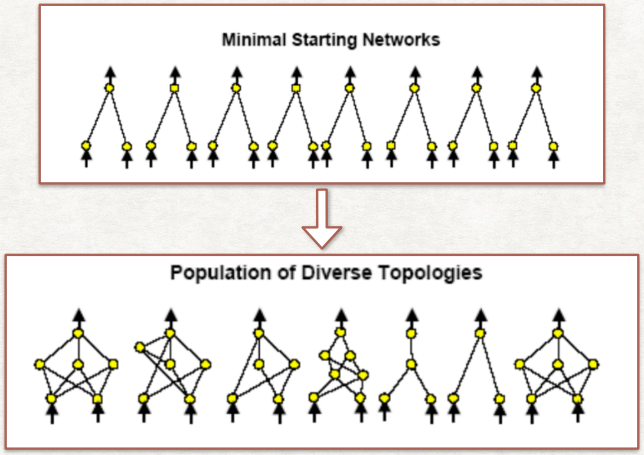
[> Evolving Neural Networks through Augmenting Topologies (Kenneth O. Stanley et al. 2002)]

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Minimality

Population starts with minimal topologies, and their complexity may increase during evolution through reproduction.

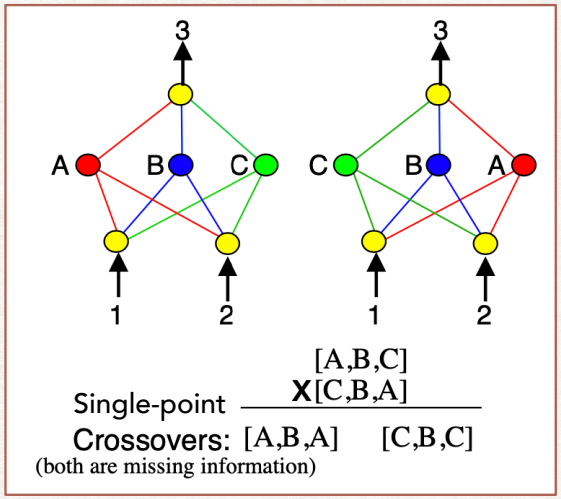
Complexity should be maintained if **useful**.



NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Reproduction: competing conventions

How do you know **which gene is which** so that you can perform crossover?
How does nature solve this?

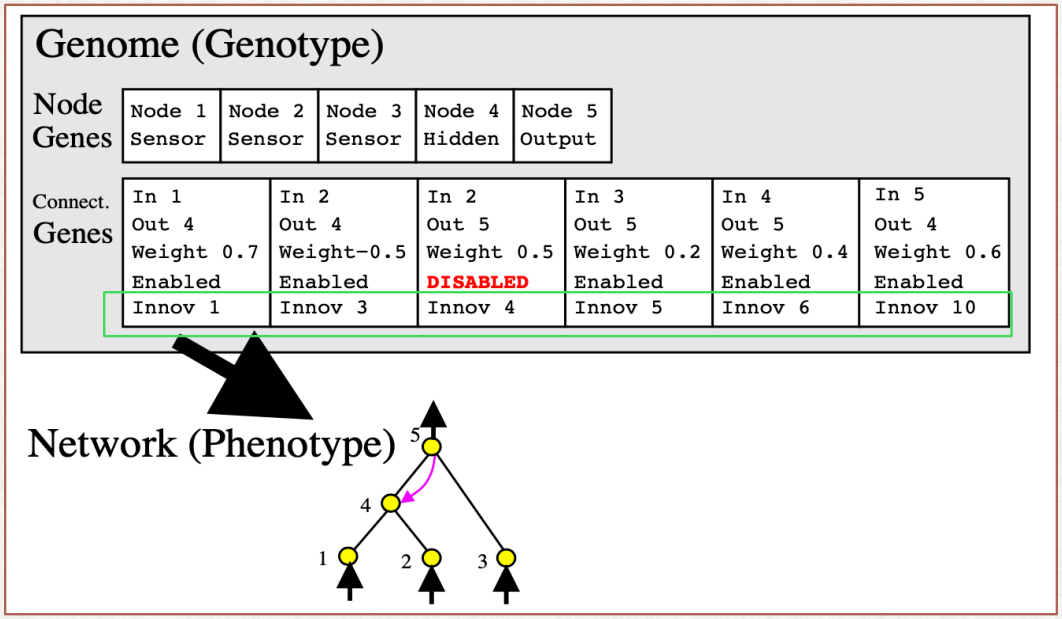


Competing conventions problem: it means having more than one way to express a solution to a weight optimization problem with a neural network.

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Reproduction: innovation numbers

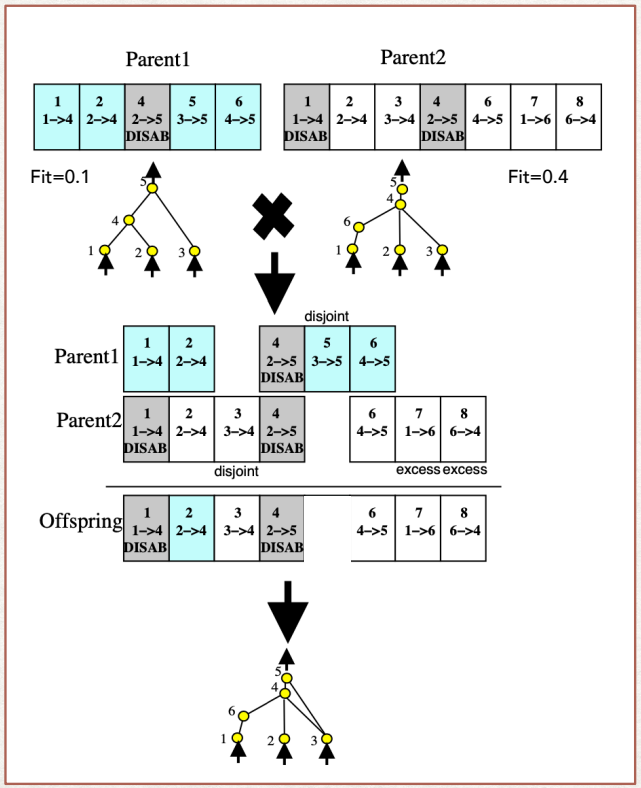
Use unique historical markers to identify genes.



NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Reproduction: crossover

Using the innovation numbers it is possible to align two genomes and perform crossover.



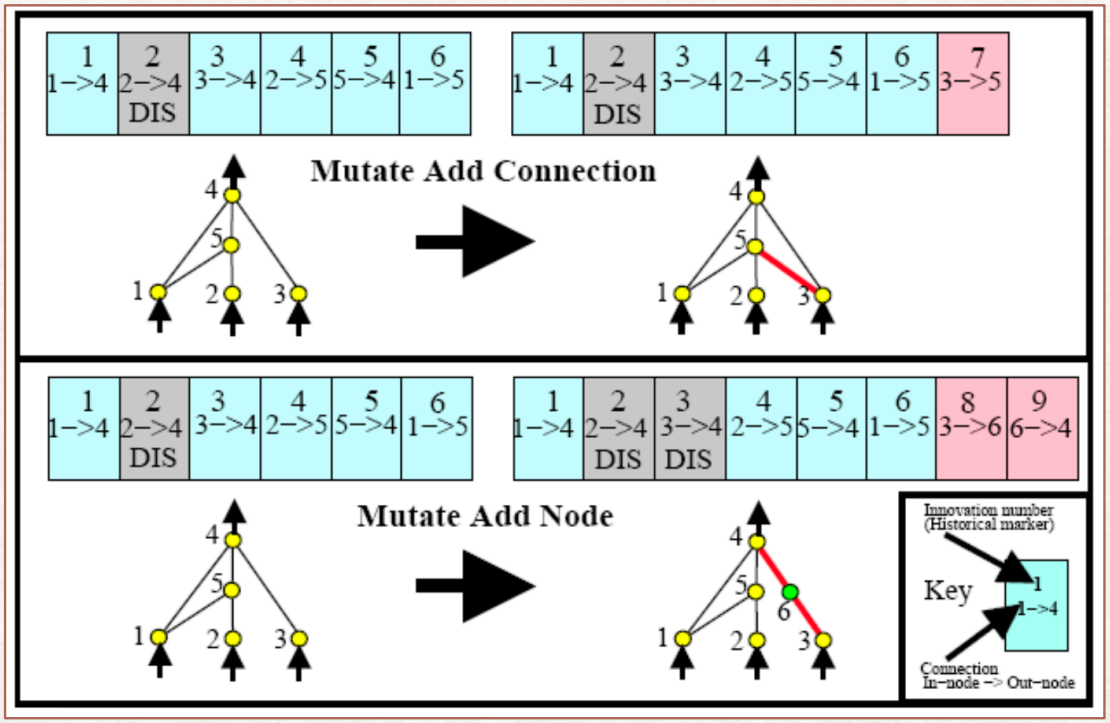
Matching genes are inherited **randomly**,
whereas **disjoint** genes (those that do not match in the middle)
and **excess** genes (those that do not match in the end) are
inherited from the **more fit** parent.

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Reproduction: mutation

Different **types** of mutation can happen, and each one has its own probability: **multiple** types could happen in the **same** mutation.

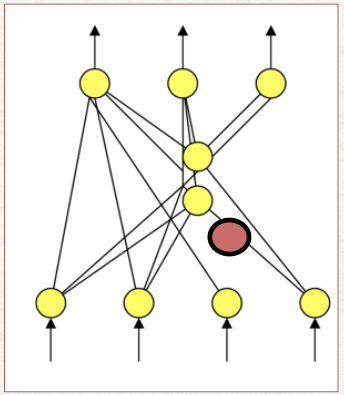
- add node
- remove node
- add connection
- remove connection
- mutate weight
- …



NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Speciation: protecting innovation and maintaining diversity

Innovations can be **disruptive**: a single extra node could make the NN behave very differently.



Perhaps, initially this new node will cause fitness to **degrade**.

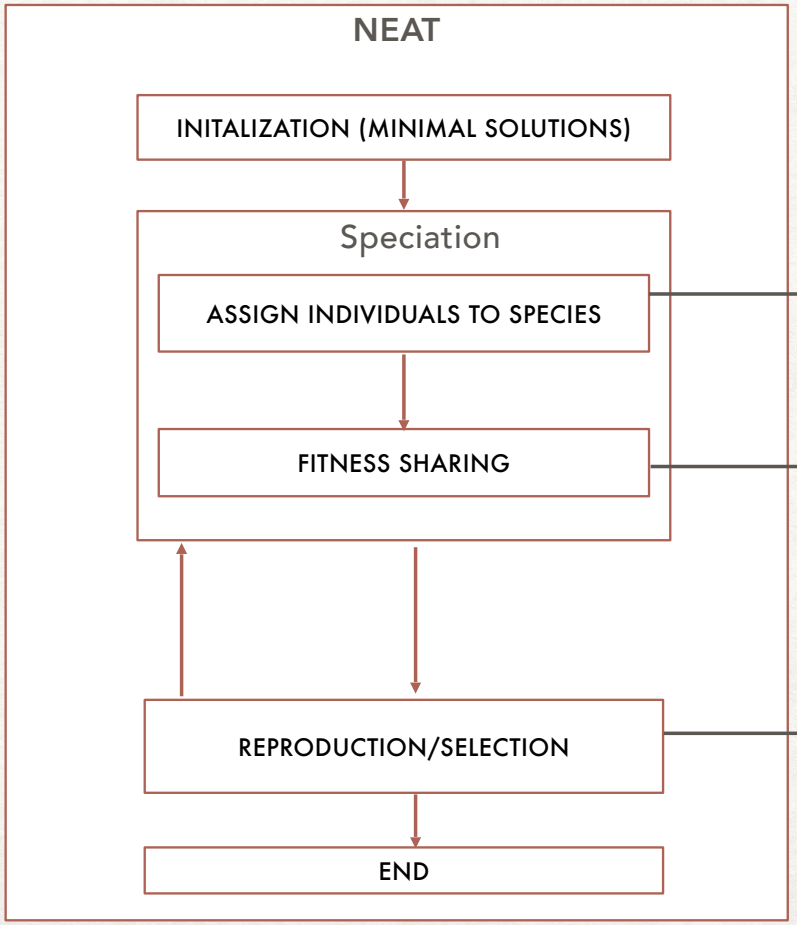
But after weights are re-optimised it may bring **benefits**.

Protect innovation by having different species: individuals **compete** only **within** the species.

Individuals **within** the species are similar.

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Full cycle



- A **list** of species is maintained.
- Each existing species is **represented** by a **random** genotype inside the species from the previous generation.
- In each generation, genotypes are sequentially **placed** into species.
- A given genotype in the current generation is placed in the **first** species in which it is **compatible** with the **representative** of the species.
- The measure of compatibility is called **compatibility distance**.
- If a genotype is not compatible with any existing species, a **new** species is created with it as its representative.

- **Fitness** of individuals is **reduced** based on species size.
- This way, high fitness species do not grow too big (taking over).

$$f'_i = \frac{f_i}{S}$$

S is the size of the species of the individual i

For each species:

- Eliminate **worst** individuals
- Create a number **N** of offspring from **random** pairs of **parents**.
- **N** is **proportional** to the sum of fitness of all parents within the species in comparison to other species.

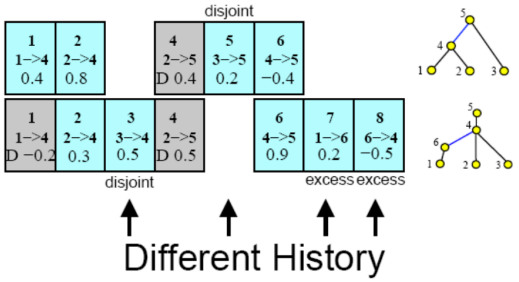
  Species with higher fitness values generate more offspring.
- **Replace** all **population** with the offspring.

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Speciation: compatibility distance

δ is the compatibility distance between two structures

The more disjoint two genomes are, the less evolutionary history they share, and thus the less compatible they are.



$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \bar{W}$$

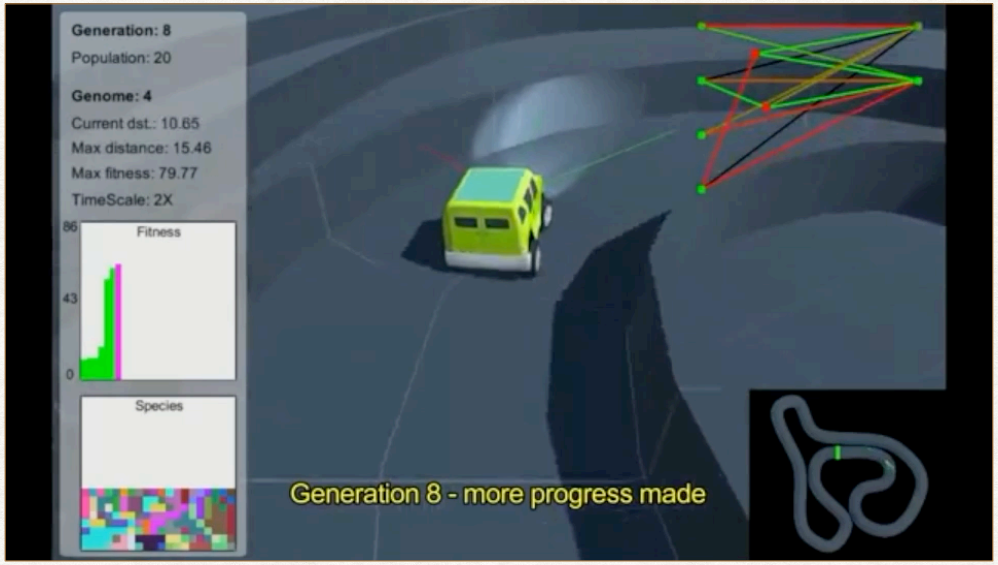
D	Number of Disjoint genes
E	Number of Excess genes
$\overline{W}$	Average weight differences of matching genes
N	Number of genes of the largest genome

The distance measure δ allows speciation using a compatibility threshold δ_t.

NEUROEVOLUTION

Applications: car driving

NeuroEvolution of Augmented Topologies (NEAT) for a car to learn to drive on the track



https://www.youtube.com/watch?v=2bW9CdFcaUI

TYPES OF ENCODING

Direct encoding

One gene per “cell” (or part) of the phenotype.

Structures are discovered redundantly, even if similar or identical.

Genotype:

Gene 1: 

Gene 2: 

Indirect encoding

Also known as generative or developmental.

Reuse of structures.

Genotype:

Gene 1: 

Gene 2: Replication/flipping “tools”

- 100 trillion connections in the human brain
- 30,000 genes in the human genome
- Only possible through highly compressed representation (indirect encoding)

CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

Evolving weights, topology **and functions**

- A developmental encoding: a neural network that generates "artefacts".

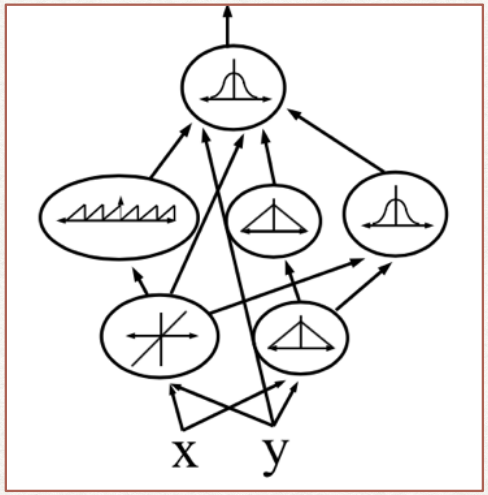
  > Benefit: a smaller search space.

- The network is a function of the problem given its geometry.

- The composition of the activation functions create patterns with regularities.

  > Benefit: object will be endowed with regularities, e.g., symmetry.

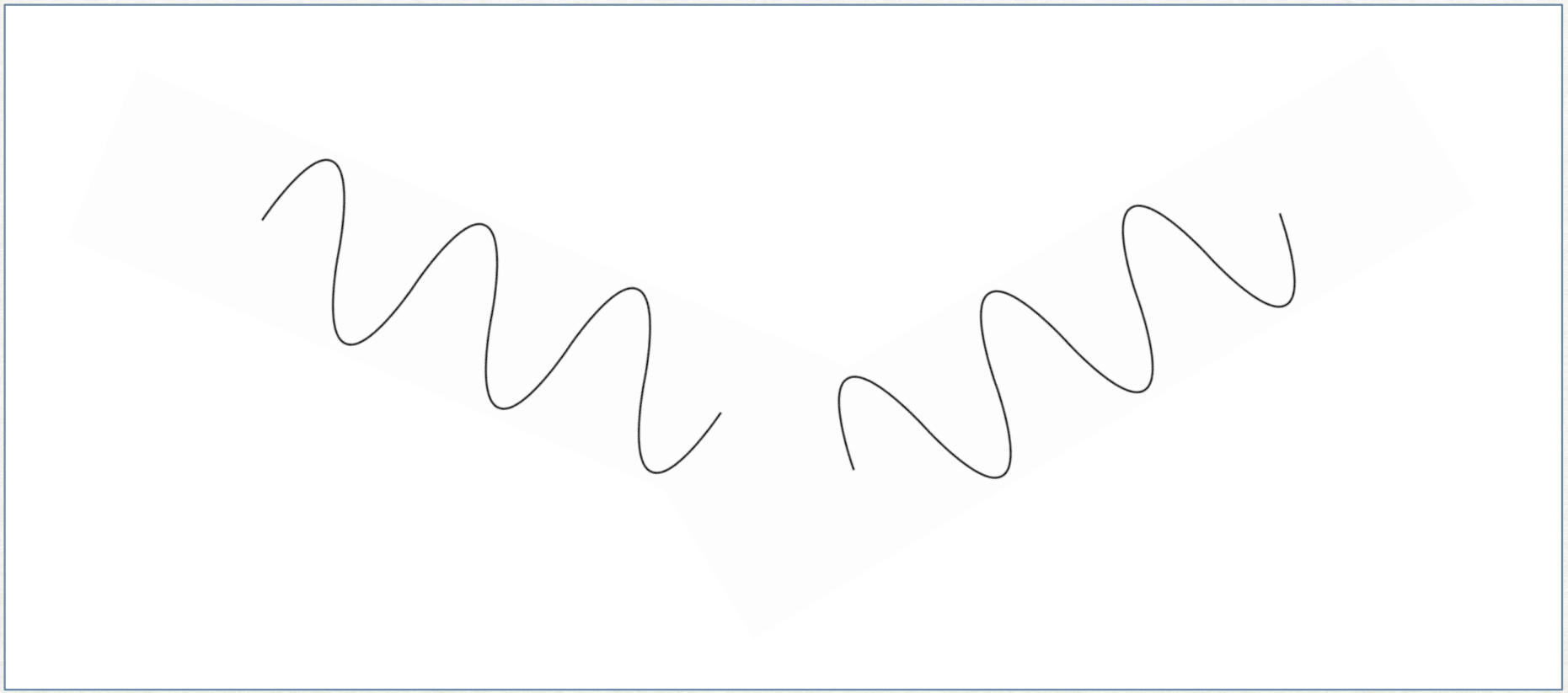
- Evolved with CPPN-NEAT: same as NEAT, but the activation functions are also evolved.
  e.g., gaussian, sigmoid, periodic functions, etc



> Compositional pattern producing networks: A novel abstraction of development (Kenneth O. Stanley et al. 2007)

CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

Example of composition: sine with gaussian

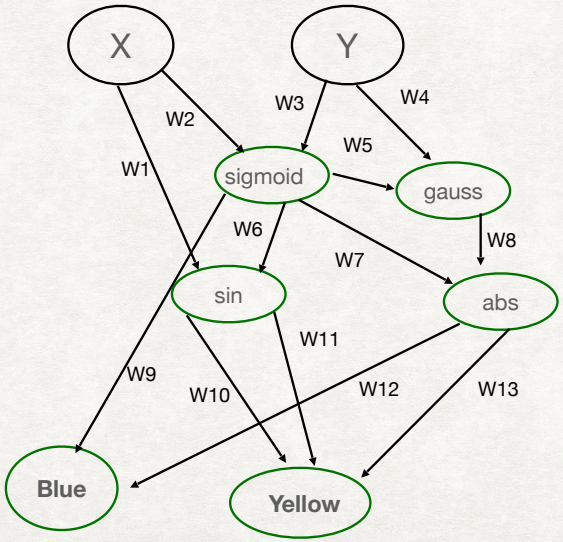


CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

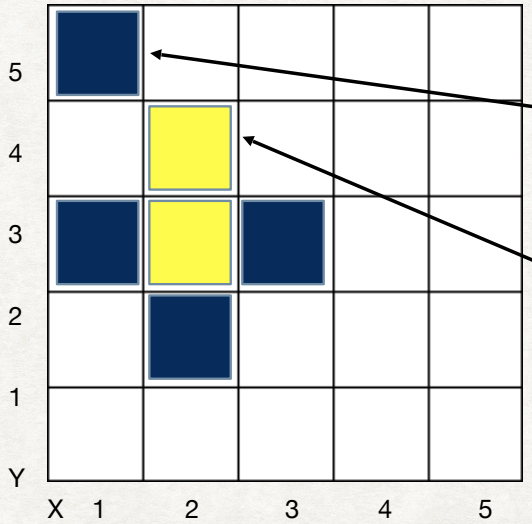
To **query** the network = means to provide **inputs** about the geometry (or context) of the problem to the network, obtaining **outputs**.

The space where queries are applied is called **substrate**.

CPPN



Object structure (substrate): a grid to be coloured



> Querying the network with **X=1** and **Y=5**

Which neutron has the highest value?
e.g., blue.

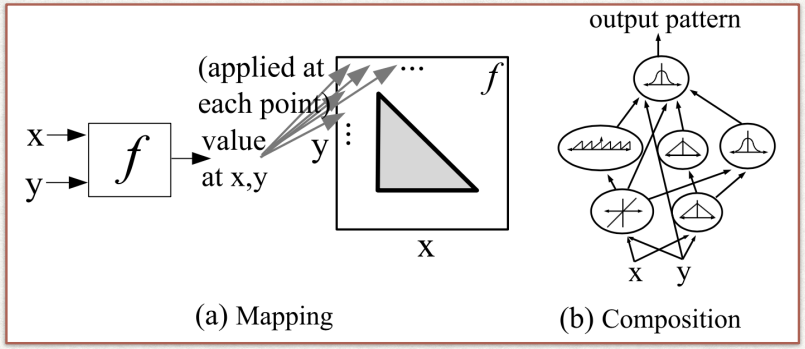
> Querying the network with **X=2** and **Y=5**

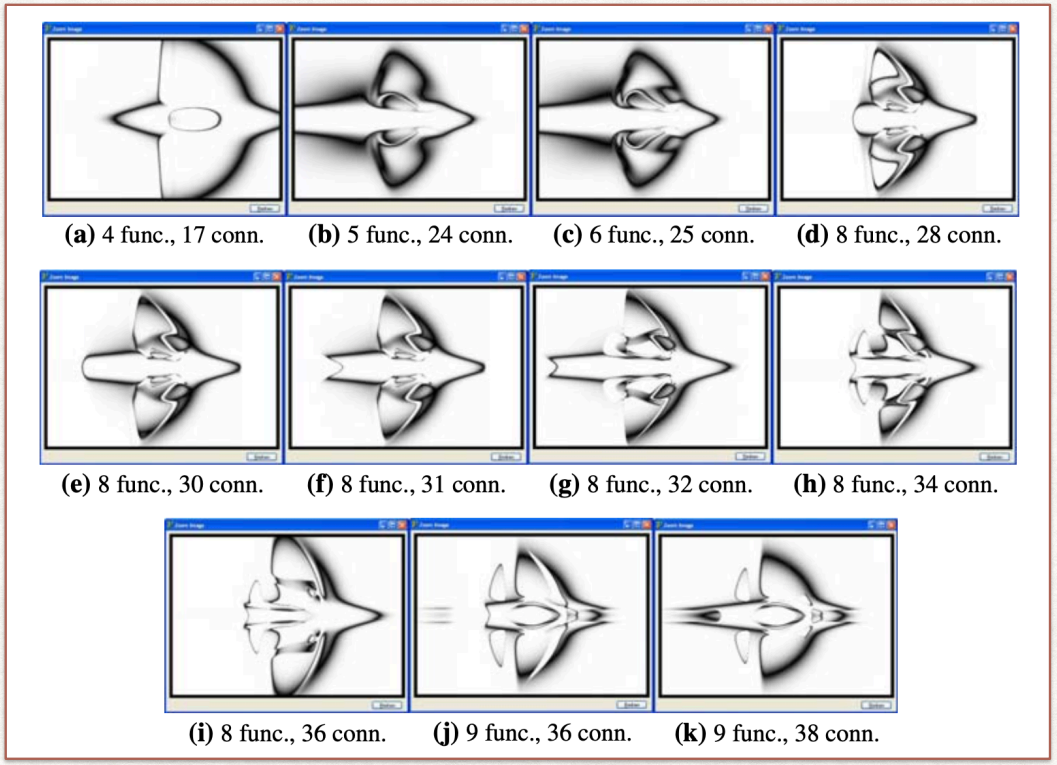
Which neutron has the highest value?
e.g., yellow.

...

CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

Drawing 2D with a CPPN





(a) 4 func., 17 conn. **(b)** 5 func., 24 conn. **(c)** 6 func., 25 conn. **(d)** 8 func., 28 conn.

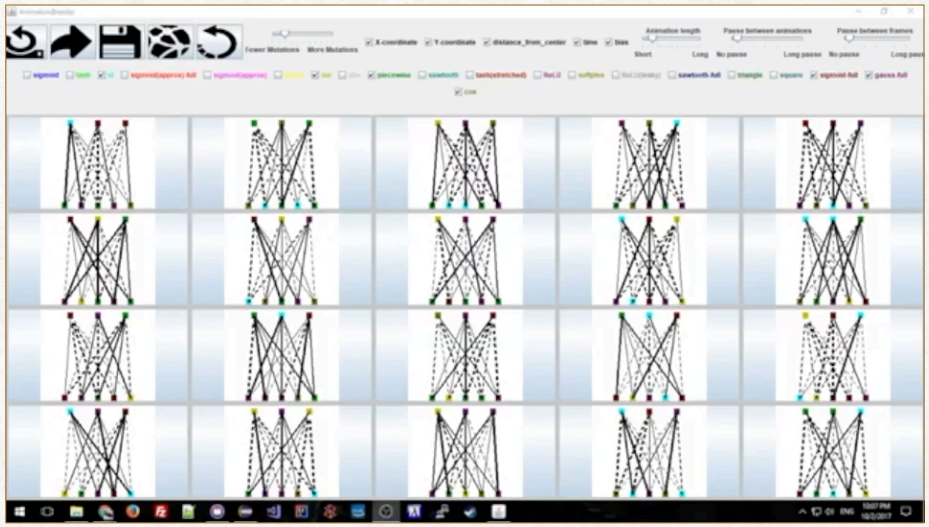
(e) 8 func., 30 conn. **(f)** 8 func., 31 conn. **(g)** 8 func., 32 conn. **(h)** 8 func., 34 conn.

(i) 8 func., 36 conn. **(j)** 9 func., 36 conn. **(k)** 9 func., 38 conn.

NEUROEVOLUTION

Applications: art

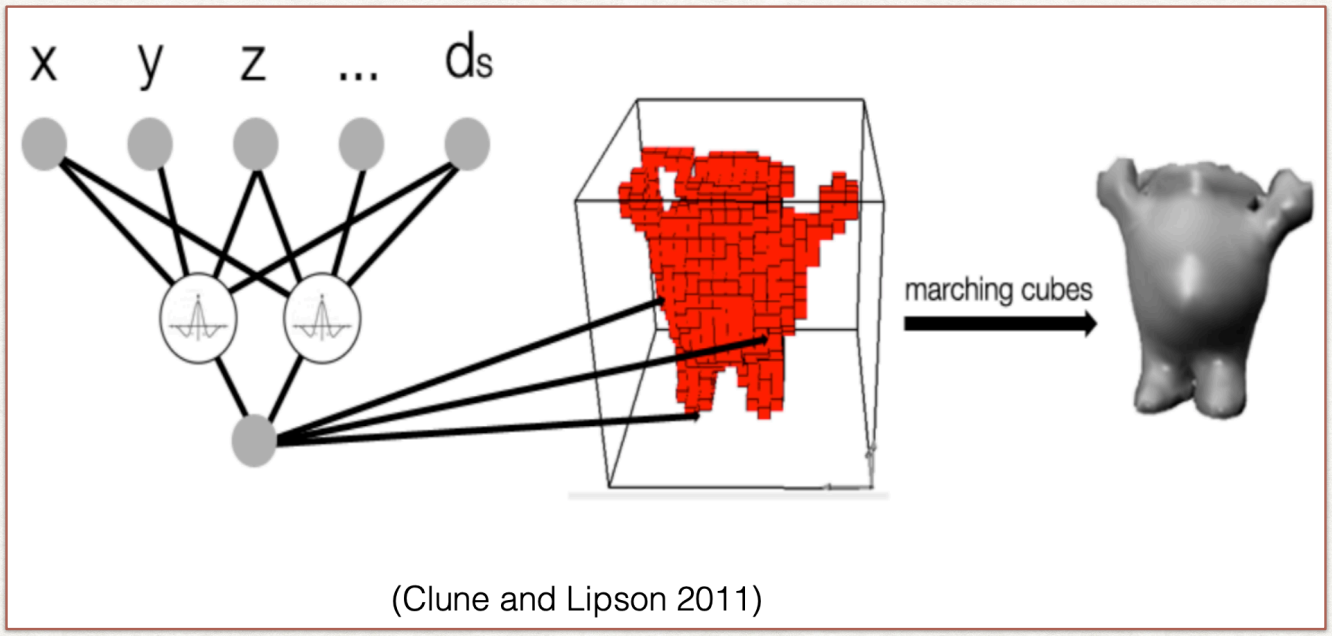
Compositional Pattern Producing Networks to create paintings



https://www.youtube.com/watch?v=xqgv0d2moa8

CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

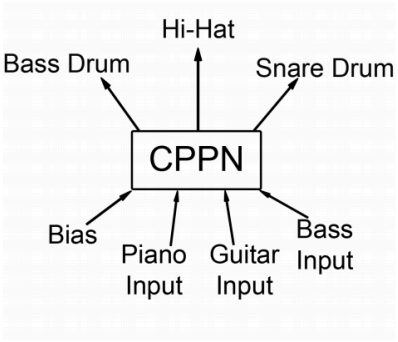
Building a shape with a CPPN



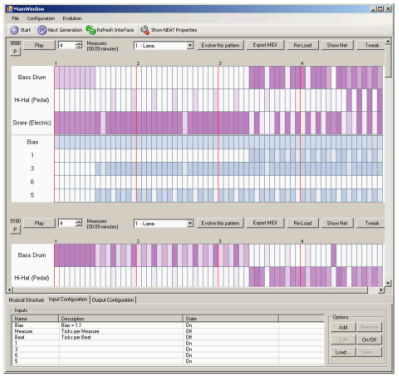
(Clune and Lipson 2011)

CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

Composing music with a CPPN



(a) Inputs and Outputs



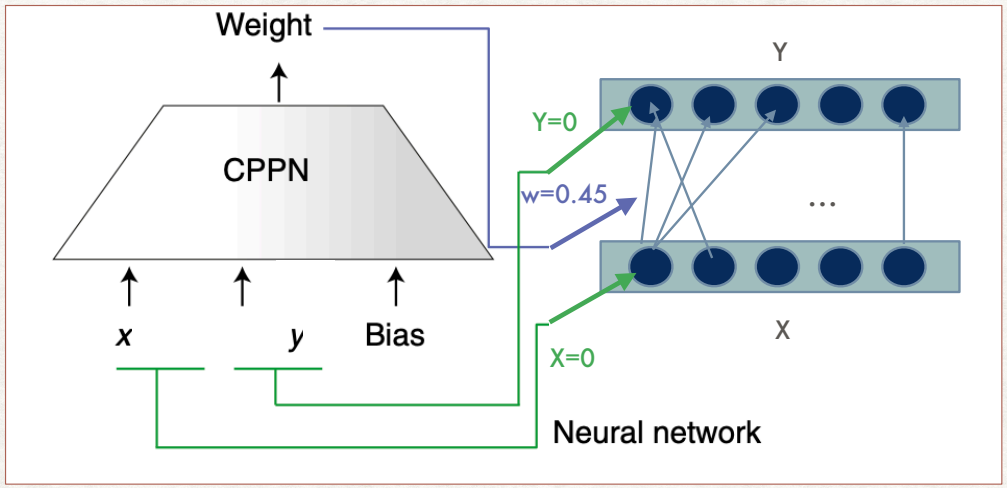
(b) NEAT Drummer Screenshot

Fig. 1. Using NEAT Drummer. (a) The user selects both a set of inputs from among the channels in the MIDI file and a set of outputs corresponding to specific drums. (b) NEAT drummer presents an interactive evolution interface where visual representations of drum patterns help the user to decide whether to listen to each candidate and then select their favorites.

HYPERNEAT

Creating a neural network with a CPPN: making a **neural network** with **another** neural network.

Use **coordinates** of neurons as input to CPPNS.

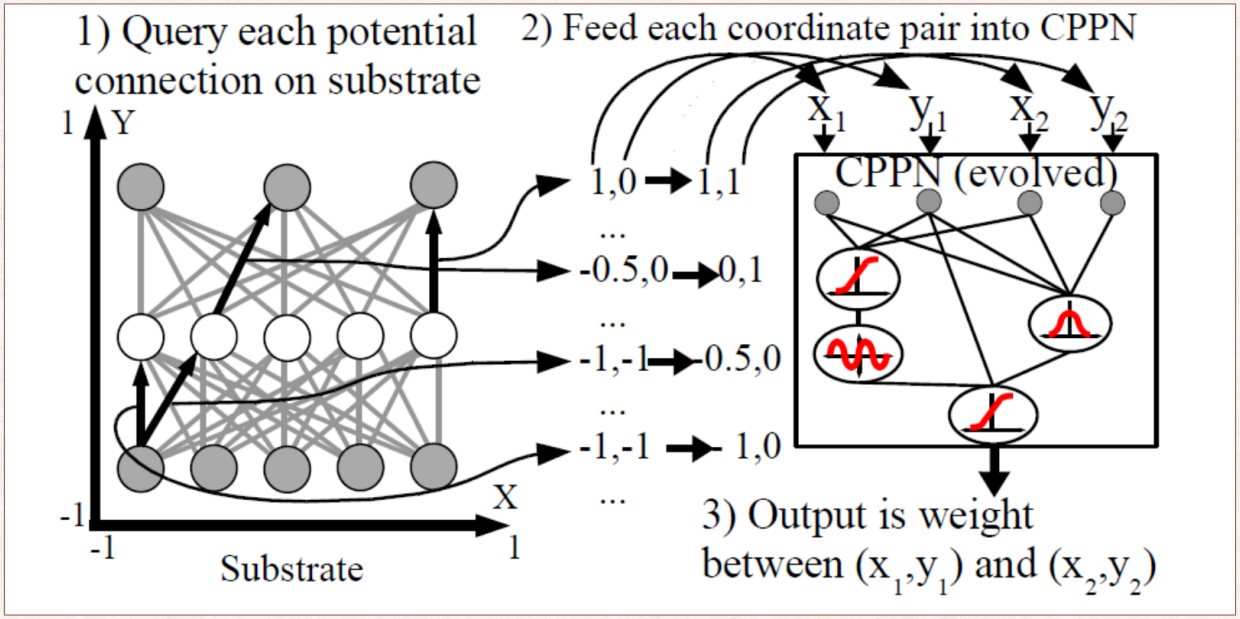


Genotype (indirect) Phenotype

[> A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks (Kenneth O. Stanley et al. 2009)]

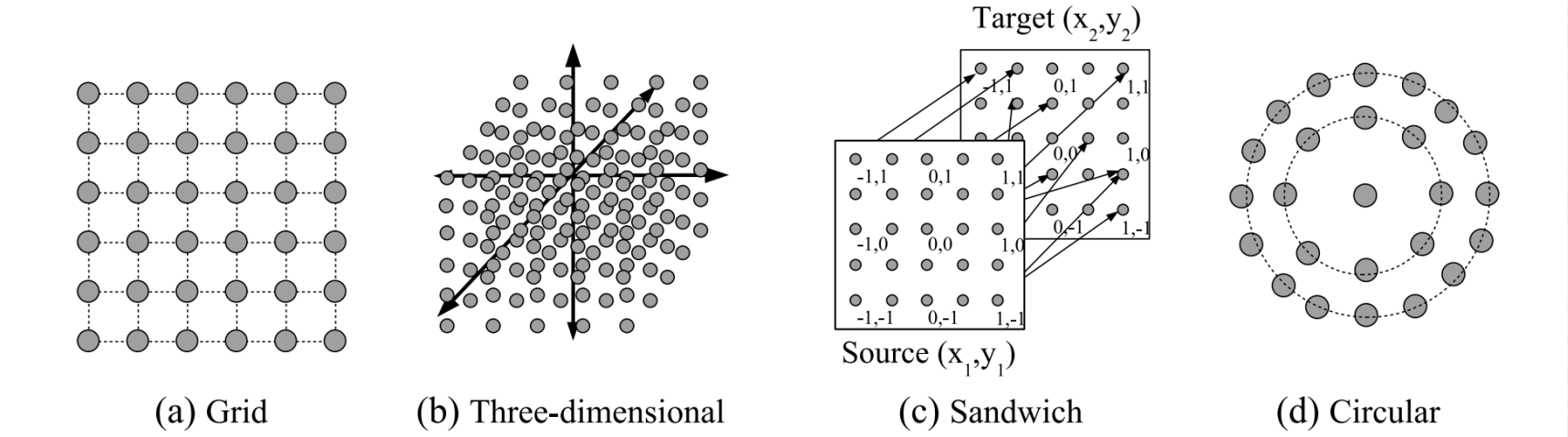
HYPERNEAT

Example with multiple layers.



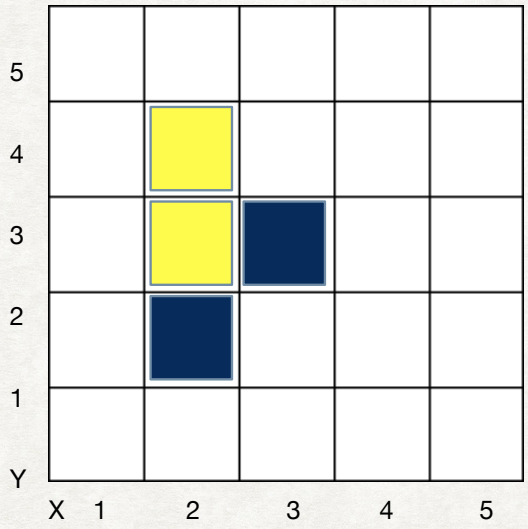
HYPERNEAT

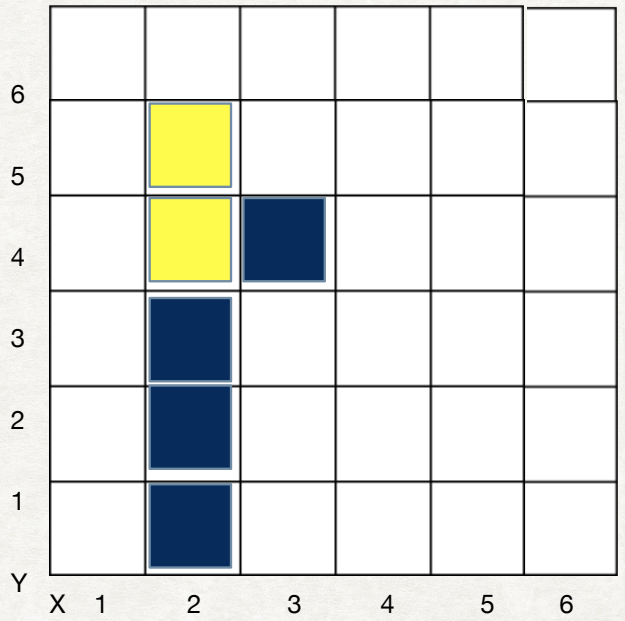
Substrate design can be adapted to the geometry of the problem.



HYPERNEAT

There is no upper bound on substrate **resolution**, that is, a connectivity concept is infinite in resolution.





EXTRA READING....

> Evolving the placement and density of neurons in the hyperneat substrate (Sebastian Risi et al. 2010)

Allows also the **evolution** of the **substrate**, so the the phenotype will not have a fixed, but an **evolvable topology**.

> A Unified Approach to Evolving Plasticity and Neural Geometry (Sebastian Risi et al. 2012)

Similar to the one above, but phenotype network has also the capacity to **learn** during life time, i.e., changing its weights.

> Designing neural networks through neuroevolution (Kenneth O. Stanley et al. 2019)

Overview about neuroevolution.

SUMMARY

> Neuroevolution can be done only on the **weights**, but also on **topology** and **activation** functions.

> Neuroevolution can be used for both **supervised** and (**semi** or **un**) supervised problems.

> NEAT can evolve networks fully, and its main principles are **minimality** and protection of **innovation** through speciation.

> A CPPN is a **developmental** representation that can encode **regularities** and can be used to generate artefacts, like drawings, shapes, other neural networks, etc.