EVOLUTIONARY NEURAL NETWORKS
4. GRADUALISM

José del Carmen Rodríguez Santamaría
https://www.ejristos.com
Bogotá, Colombia

May 20, 2025

Abstract

We investigate the possibilities of evolution in building Neural Networks (NNs) that recognize images. In previous articles, we have worked with human designed NNs that were noise tolerant boolean circuits. The task of Evolution was to specify some parameters. Our results were very humble, a behavior that is possibly due to the saltationist, discontinuous character of those circuits. In this article, we accept the challenge of engineering and studying gradualism, under which close outputs are produced in reaction to close inputs.

Contents

1 THEORETICAL FRAME 3

1.1 Gradualism vs Saltationism 4

2 OUR PURPOSE 8

3 IMPLEMENTATION OF GRADUALISM 9

4 CONCLUSION 18

5 Answers to exercises 18

6 THE BIBLIOGRAPHY 25

1 THEORETICAL FRAME

Our general aim is to use Evolution to design Neural Networks (NNs) that classify images. We work with toy models to get insight and to allow calculations to be done in ordinary computers. Our images consist of patterns that are drawn in a 3×3 grid whose pixels are white else black and that therefore can be described by binary strings 9 bits long. These are called **bitmaps**.

Each bitmap is classified by the criterion of the closest neighbor according to the BBB-distance as a deformation of one of two archetypes, an image of 1 and another of 0. The BBB distance is the Bit-By-Bit distance which counts the number of places where two binary strings differ. Example, the BBB distance between 000 and 010 is 1, and 101 is closer to the first pattern than to the second.

Our perceptrons are indeed generalized boolean gates that are tolerant to noise and variability. We proved in our *Intro* that a NN with just two perceptrons suffice to recognize the half of 512 possible patterns. Evolution adjusted separate sets of diverse parameters of that NN. (Rodriguez, [8] 2024a). Let us remark that our generalized boolean gates are full-force creators of jumps, of discontinuities, because two almost equal inputs can be separated by the threshold and therefore the responses are extremely different, in one cases the perceptron fires but not in the other.

Since with two perceptrons we were able of matching the half of cases and without mismatches, we aspired to get perfection with the addition of some few pairs of perceptrons. What we found is that perfection, to correctly classify all 512 bitmaps, is attainable but up to an exaggerated cost of nearly 250 perceptrons. Anyway, we also found that Evolution has some power to simplify the NN (Rodriguez, [9] 2024b).

We also applied the old strategy in Computing Science of *Divide and Conquer*, which divides a complex problem in sub-problems, whose solutions shall be combined to generate a solution to the original problem (geeksforgeeks, [4] 2024). Thus, we divided the 3×3 grid in 3 rows. This causes that each bitmap is divided in three sub-bitmaps that must be recognized by separated circuits, all of which shall be joined to produce a final classification of whole bitmaps. But we failed: the divide and conquer strategy produced poorer results than the initial procedure. (Rodriguez, [10] 2024c).

1 *Exercise.* *By studying a toy model, we arrived to the conclusion that NNs with a large number of inputs that are targeted to large patterns are cheaper and have more recognizing capabilities than conglomerates of small NNs that point to subpatterns. This was proposed as an explanation of why neurons are obsessed with large numbers of inputs or dendrites (Rodriguez, [10] l.c., 2024c). Do you consider that this conclusion is firm and shall not be revisited anyhow? Answer*

Let us enter now to discuss the theme for the present work.

1.1 Gradualism vs Saltationism

Every human being knows that a change with few details is easier than another with a large number of specifications that work in toto. Evolution is charged with the very same idea. In this regard, the vision of Darwin is worth a cite. Let us see, in first place, how he tries to reduce complex tasks to a series of slight subtasks - a strategy that in Computer Science is called *divide and conquer*-, and that, according to our owns results (Rodriguez, [10] l.c., 2024c), is not unconditionally correct:

To suppose that the eye, with all its inimitable contrivances for adjusting the focus to different distances, for admitting different amounts of light, and for the correction of spherical and chromatic aberration, could have been formed by natural selection, seems, I freely confess, absurd in the highest possible degree. Yet reason tells me, that if numerous gradations from a perfect and complex eye to one very imperfect and simple, each grade being useful to its possessor, can be shown to exist; if further, the eye does vary ever so slightly, and the variations be inherited, which is certainly the case; and if any variation or modification in the organ be ever useful to an animal under changing conditions of life, then the difficulty of believing that a perfect and complex eye could be formed by natural selection, though insuperable by our imagination, can hardly be considered real. (Darwin, [2] 1857, page 186).

What could happen when the aforementioned strategy does not function? He recognizes that this would be mortal to his theory but he considers that that is not the case in Biology:

If it could be demonstrated that any complex organ existed, which could not possibly have been formed by numerous, successive, slight modifications, my theory would absolutely break down. But I can find out no such case. No doubt many organs exist of which we do not know the transitional grades, more especially if we look to much-isolated species, round which, according to my theory, there has been much extinction. Or again, if we look to an organ common to all the members of a large class, for in this latter case the organ must have been first formed at an extremely remote period, since which all the many members of the class have been developed; and in order to discover the early transitional grades through which the organ has passed, we should have to look to very ancient ancestral forms, long since become extinct. (Darwin, [2] 1857, page 189).

Let us notice that Evolution, being a blind and mindless watchmaker (compare with Dawkins, [3] 1986), is in the insight of Darwin charged with a human idea about complexity. Is that correct?

Yes, absolutely. We can justify such a view as follows:

We as well as Evolution execute tasks algorithmically, following verbal instructions in tandem: do this, then that, then this other,... Officially, *the Genome is Software and Evolution is a Software Developer* . Thanks to Molecular Biology, this insight rests on firm footing. The Evolutionary Theory (ET) now reads: *Evolution is the software developer that is responsible for the appearing of all species* (Rodriguez, [6], 2010). Adopting this stand, each branch of an Evolutionary tree is a list of expected modifications in tandem that Evolution presumably executed when passing from species to species until the final one, which is represented by a leaf.

The insight of *the Genome is software* and of *Evolution as a software developer* allows to devise a straightforward falsification of the Evolutionary Theory if only we accept the criterion that was established by Darwin:

Most children are perfect at birth, even in traditionally small populations. This is also true for every population of living beings. From that, the following implication is immanent: such a high degree of perfection as that we see in almost all living beings has been preserved in spite of mutation because the corresponding genetic information amounts to giant packages of instructions that cannot be modified. The consequence of one single mutation is that essential functions fall apart. For mammals, an abortion is produced. This kind of fatal failure is also typical in software systems developed by human beings. In Biology, the unmodifiable genetic core is responsible for the notion of species. Side by side with such a unmodifiable core, there are too much space for change and that is why we are all so different from

one another. Additionally, there are also space for much change in tandem and so, there are many "proofs" of the ET, for example, the Evolution of Covid viruses shows that some parts of their genome are plastic (Amoutzias [1] 2022)

To pass from rhetoric to Science, we must make experiments that lead to predictions to next compare them with observations. Computer experiments with Evolution are called *Genetic Algorithms*, and when Evolution is used to design software, we have *Genetic Programming*.

The two main results of our studies in Genetic Programming are:

- Evolution can synthesize software.
- Garbage is always produced by immense volumes that augment when the complexity of the used language grows. (Rodriguez, [11] 2021d, [12] 2022e) .

What is garbage in software development? It is software that produces nothing else defective products. In our programs, our simulations of Evolution, garbage is reported as the cost in individuals to arrive until the current generation, specially when perfection has been attained. Thus, **we have a mandatory prediction**: *If life on Earth is the result of Evolution, then the fossil record as well as our own body must be filled in imperfections, malformations and malfunctions that must be visible from birth.* **This is contrary to observation.** For instance, I am 71 and, in spite of all my troubles, I wonder at the great perfection of my mind and senses. That is why the ET is for me obviously false (compare Rodríguez [7], 2021a). The Reader is invited to make the corresponding judgment over her or his own body and faculties.

In short: The ET is false, but Evolution is real. And Applied Evolution is as real as we want. In particular, we have shown that Evolution can help in the design of NNs. As yet, we only have shown

that Evolution can find good values for separate sets of parameters of a toy NN.

Our NNs have been Cascade Circuits of Fault Tolerant Boolean Gates, a specific type of perceptron or artificial neural cells. We have worked with circuits with one or more layers that recognize patterns in the presence of noise and variability. In spite of the fact that we have worked with possibly the simplest toy model, we have frequently run short of memory and patience. Why? Our guess is that changes in the input or in parameters may have the output to go in jumps and can cause disproportionate change in the fitness. Our Evolution is saltationist. So, Evolution cannot gather small improvement. It is straightforward to infer that to gather small improvement, we must resort to Gradualism, a type of Evolution in which small changes in the input produce small change in the output and fitness. Gradualism also requires low mutation and recombination rates, a large enough population size, and a fitness landscape without too much picks.

2 OUR PURPOSE

We align ourselves along the extended belief that Evolution goes swiftly when allowed changes in the genome produce soft changes in the fitness function. If Gradualism is not implemented, Evolution would be curved, braked. Why? Under gradualism, changes are small and result from few details, while in saltationism, fitness remains in ecstasy for too many generations, but if time is enough, a favorable large change with numerous specifications that work in toto may happen. But the demanded lapse of time maybe longer than the age of the Universe. Thus, **Our purpose for this article is to engineer gradualism for the specification of the parameters of a NN and to test what it gives from itself.**

3 IMPLEMENTATION OF GRADUALISM

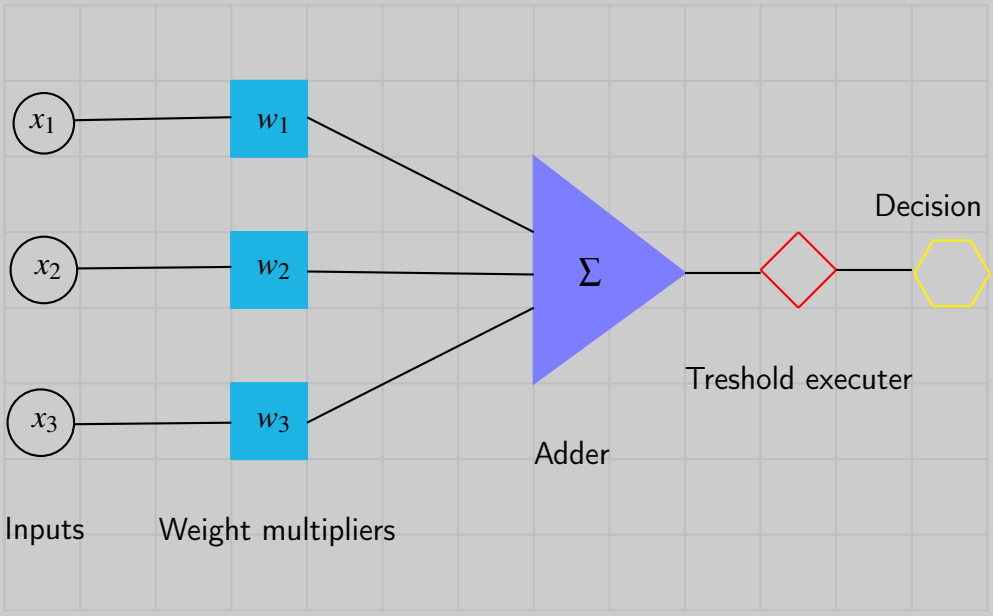
Our task is now to detect any trace of jumps in any part of our perceptrons and to implement soft changes everywhere. From our *Intro*, we know:

A perceptron can naturally be interpreted as a machine that makes decisions. See the figure 1. According to our implementation until here, it receives binary inputs, x_i, that say whether a certain factor exists, 0 when not, 1 for yes. Each input is multiplied by a number that represents its importance, w_i. The results of the multiplications are added together to get $S = \sum w_i x_i$. To gain stability against noise and variability, we add a positive value, b. We get $S = \sum w_i x_i + b$ If the sum S surpasses a certain threshold T, the decision is affirmative else it is negative.

In short, we get $S = \sum w_i x_i + b$ as the input to the threshold executer that checks the value of S against the threshold T: if $S \geq T$ the decision is affirmative, else it is negative. In the literature, the shifting value b is called the **bias**.

Jumps are synonymous of discontinuities and presumably responsible for the subsequent failure of Evolution:

- The presence of absence of a stimulus is dichotomous, to be else not to be. This means that the pixels of our graphics are white else black.
- To simulate boolean gates, the weights are 1 else 0.
- The bias is an integer value.
- To reproduce boolean gates, an additional bias can be assigned to each factor. It can take on the values 0 else -1.
- There is a threshold that responds quite different to almost equal inputs.



j

Figure 1: This is a perceptron. It is an abstraction of a neuron that can naturally be interpreted as a machine that makes decisions. Suppose we need to decide whether to continue reading or not an article. We have as input to the perceptron the presence or absence of some factors, x_i, say, THE READING IS SEEMINGLY VERY IMPORTANT TO BUILD A COSMOVISION, and IT IS INTERESTING. Factors have different weights w_i to influence the decision. So, inputs are multiplied by their weights and their multiplied values are added together. To enhance the stability of the perceptron against noise and variability, a constant b is added to the sum, to get $S = \sum w_i x_i + b$. If the sum S surpasses a certain threshold T, the decision is affirmative else it is negative. Thus, the decision is affirmative if $S \geq T$, otherwise, it is negative. We must attack the following characteristics to implement gradualism : 1) the values of the factors x_i are dichotomous, 1 = present else 0= absent, 2) the weights are imposed by Boolean gates, 1 else 0. 3) The bias is an integer number, 4) The decision $S \geq T$ is YES else NOT. 5) We conjecture that the threshold is possibly the most important source of jumps.

- A decision can be YES else NO that we prefer to interpret as YES else UNKNOWN.

Thus, our perceptrons are fulbright jump-creators.

Remedies to try out:

- We have worked with bitmaps, digital representation of a drawing, whose pixels are white else black. Now, we might try gray values. The pixels of such a graphic are represented by numbers in the interval [0,1] to represent diverse tones of gray ranging from 0 for white to 1 for black.
- Instead of integers, the weights take on real values, i.e., on the interval $[-\infty, \infty]$. For practical reasons and to begin with, we will replace the whole real line by the interval $[-99, 99]$ and we will work with 2 significative ciphers. In that way, a number will look like, say, -13.24. The sign will be encoded as the first digit, if this is even, the sign is plus, if this is odd, the sign is minus. For instance, the string 12345 represents the number -23.45 but the string 82345 represents the number 23.45. The decimal point will not be encoded because it will always be in the same site. This means that factors are not simply accepted else rejected but can have relative weights that can be interpreted, for instance, from rejection to affection with every degree of variation in within.
- The bias takes on real values instead of integers.
- Additional bias are not included, here they are redundant.
- We have a great problem with the threshold that causes the NN to respond quite different to 4.9 than to 5 because the NN can respond only YES else NOT. We suspect that this is the major cause of chaos that destroys gathered favorable change and

turns Evolution into a frustration because of the over multiplication of demanded resources, CPU time included. Our solution follows the TRIZ rule: *the best solution is no solution*. So, we will investigate what happens if we abolish thresholds and decide that a *cell or perceptron fires as response to a given input when its output is the largest in comparison with all perceptrons or cells in the same layer, and we also define that the other cells do not matter so, their responses are taken as* 0*, or NOT-KNOWN*. With this rule, cells cannot contradict one another. This rule was inspired by the ReLu function of the ordinary theory of NNs (Nielsen [5] 2018)

This form of Gradualism is implemented in program `EvolNNGradualism.html`.

2 ***Exercise.*** *Run program* `EvolNNGradualism.html` *to test the conclusion of the Author: this form of gradualism resolves all problems with the help of just two perceptrons. It allows Evolution to reach perfection, 512 matches out of 512 options, under no mismatch, usually before 2000 generations with a population size of 100, and to maintain it indefinitely. Moreover, used resources are kept withing boundaries.* Answer

We have worked with saltationists models and we have found that the cost of going beyond mediocre solutions is intolerably high. And here, out of the blue, comes gradualism and rapidly exhibits perfection that is moreover maintained for eternity.

3 ***Example.*** *The claim that gradualism is miraculous is a serious one. How can you make sure that your software is correct and that this pretension is not a false illusion?*

We work with a toy model, a simple problem that can be studied with an ordinary PC. Results clearly show that Evolution is not magic but real. One sees how it gain fitness in rather small steps

that may happen after long waiting times, until it reaches perfection, which with too much difficulty we were able to smell in the previous saltationist setting. We consider that this behavior is so special that we bet that the software is correct and that it simply does what it promises.

Anyway, we consider that software-proofing and debugging are sacred-duty tasks that must be directly accessible to everyone. So, one can see various boolean variables after line 291 in program `EvolNNGradualism.` that allow to choose specific methods or procedures in order to print their outputs. One can see in that way the output of many methods and check them.

The implementation of gradualism is what most matters for this article. So, a specific test was devised whose result can be seen in a separate text-area TEST. The implementation of this test appears in program `EvolNNGradualism2.html.`

4 ***Challenge.*** *Resolve once and for all times the following complaint: the present implementation of gradualism needs a whole program to test its correctness. Is that transparency? No, by no means. Do something better.*

5 ***Example.*** *Our Gradualism is incomplete because it needs to be told what the maximum value is between a pair of real values. This was done with a few lines of TypeScript code, which corresponds to receiving help from a Turing machine. To complete our Gradualism, we will show that an RN can find the maximum value between two real numbers. In this way our RN becomes a full gradualist.*

Let us suppose we have two numbers, x and y. To know whether x is the max between two, we subtract $x-y$. If the result is positive, x is the maximal value. If it negative, x is not the maximal value. This can be rewritten as $x-y \geq 0$ that is the direct determination of a perceptron whose weights are 1 and -1, its bias is 0 , and the threshold is 0. Let us denote this perceptron as the subtractor $X-Y$.

An NN that finds the maximal value between two numbers, x and y, is built as follows: there are two inputs, x and y, and two output lights, X and Y. The first fires when x is the max, the second fires when y is the max. In between inputs and output we have two perceptrons, the first is $X-Y$, the second is $Y-X$. Each subtractor has two input pins and one output. Connections are shown in figure 2.

6 *Challenge.* *Design an NN that finds the maximum among three real numbers. Generalize to n numbers.*

7 *Example.* *The ingredients of our implementation of gradualism are: to convert all parameters into real variables and to burn out the threshold. We have contended that the burning of the threshold is the single most important factor that propels gradualism. To test this claim, we build a NN with real parameters but with a threshold that we put at zero. Next, we study it to see what happens. Our solution has been packed in program* `EvolNNGradualism3.html`. *Let us explain why we can put the threshold at 0:*

For each perceptron, we calculate the sum S given by $S=\sum w_i x_i + b$. If the sum S surpasses a certain threshold T, if $S \geq T$, the decision is affirmative, otherwise, it is negative. Thus, we are dealing with the inequality:

$\sum w_i x_i + b \geq T$

where b and T must be specified by the Evolutionary Process. This is equivalent to

$\sum w_i x_i + b - T \geq 0$

Since b and T are unknowns that are unrelated to the other parameters, we can rename the unknown $b-T$ as b, to get

$\sum w_i x_i + b \geq 0$

That is why we posed our threshold at 0.

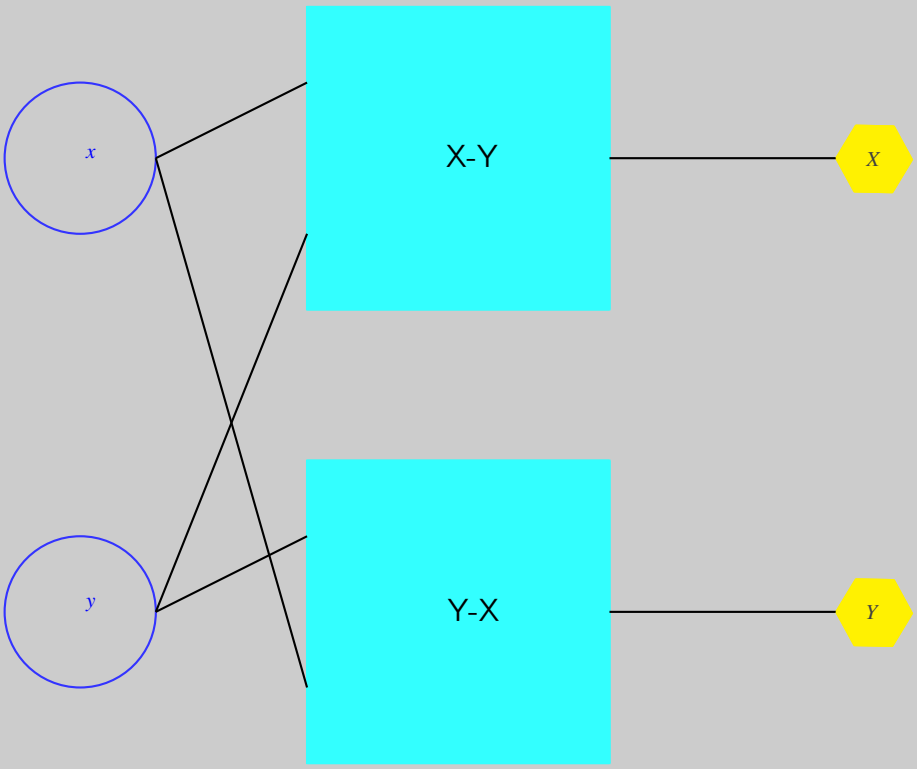


Figure 2: This NN finds the maximum between two numbers x and y. The NN has two perceptrons. The upper perceptron must fire when x is the max, the lower when y is the max. The two perceptrons are of the same nature, they subtract the number incoming to the lower pin from that incoming to the upper pin and compares the result with the threshold 0. Connections are such that the upper perceptron executes $x-y$ while the lower $y-x$ and compares the result with the threshold 0. So, the upper perceptrons fires when the subtraction is positive because x is the greater, so $x-y \geq 0$ while the lower when y is the max. When the inputs are equal, the two perceptron fire.

What did we get?

Program `EvolNNGradualism3.html` was run during 20000 generations with a population size of 100. JavaScript was overloaded with the printing task due to the need to report results during 20000 generations. The result of the experiment was simple: Evolution was unable of making one single match, no one single positive result was found. This was unexpected for us and doubt arose: Is the program correct?

This program is correct because it is a slight variation of program `EvolNNGradualism.html.` that shows how gradualism works with flying colors when there is no threshold. To expand the window of observation, we did some changes: the mutation rate was moved from 0.0001 to 0.1. Failure. The mutation rate was moved to 0.3. Failure. The program was run during 100000 generations. Failure. The encoding was changed from 12.34 to 123.4. Failure. These changes were chained. At last, we run the program during 100000 generations with mutation rate 0.0001 and population size 100. The mutation rate was so low because according to previous results, a low mutation rate is necessary to reach and keep perfection. Failure.

We interpret this recurrent failure as a proof that the threshold is the cause of all troubles.

8 *Exercise.* *Program EvolNNGradualism3.html. was unable to find a satisfiable valuation of the parameters of an NN with real parameters. Why? We see a disjunctive: It is because no satisfiable valuation exists or because its probability is too low. In consequence, prove that the NN of program EvolNNGradualism3.html., which has two perceptrons of real parameters and a threshold, cannot be satisfied anyhow, i.e., there is no valuation of parameters with at least one match. Else, define a set of values of its parameters such that the NN finds at least one match, i.e., it can correctly classify at least one bitmap but with no mismatches.* Answer

9 *Exercise.* *We have said that the most important obstruction to gradualism is the threshold. Thus, we have despised the role of encoding the parameters as real numbers. Test this negative belief.* Answer

10 *Exercise.* *Resolve the following complaint that is more like an attack: Gradualism means gathering of small improvement in fitness due to small change in the genome. Gradualism seems to guarantee that Evolution will reach a local optimum rather fast. But here, the string* `01234` *represents the real number* 12.34 *while the string* `11234` *represents* -12.34*. Hence, minor changes in the chromosomes cause large changes in the value of the associated real numbers that may cause violent change in the fitness or score function when perceptrons have a threshold. As a consequence, Evolution does not function when the threshold is included.* Answer

11 *Exercise.* *Clearly show that our Gradualism have nothing to do with Natural Evolution nor with the Evolutionary Theory of the Origin of Species.* Answer

12 *Exercise.* *Justify else refute the following reasoning : All experiments, ours and from other Authors, deal with directed Evolution, when a goal is present. Since these have nothing to do with Natural Evolution, Science has nothing to say in regard with the falsification of the Evolutionary Theory. In short, the Evolutionary Theory is unfalsifiable.* Answer

13 *Graduation.* *We have learned that the encoding with format* 12.34 *helps the cause of gradualism but not too much. We thought at the beginning that satisfiable valuations are few. But, we found next that the encoding* 123.45 *was much better because beginning with a satisfiable valuation, Gradualism can make a significant improvement. We infer from these results that satisfiable valuations are many but they were not found because of the encoding with format* 12.34 *lacks sufficient expressive power. Prove this, i.e., find an encoding that allows Evolution to swiftly risen up to perfection.*

4 CONCLUSION

A toy Gradualistic Neural Network functions just as expected by Darwin: Evolution gathers small progress until stable perfection, bit by bit, but maybe after long waiting times. We have also proved the intuitive claim that the threshold of perceptrons is the main cause of troubles for Evolution. In fact, the threshold is a creator of chaos because it causes the system to respond very different to almost equal inputs. Warning: Directed Evolution has a precisely defined goal and reward, but Natural Evolution has no purpose, no goal, no predefined reward. As a consequence, one cannot extrapolate results from simulations to Nature. Nevertheless, Goalless Evolution is slower and produces more garbage than Directed one so, we are right every time we say that our simulations falsify the Evolutionary Theory due to the immense volumes of garbage, which when referring to software development are represented by no output or defective products, i.e., extinctions, malformations and malfunctions as in the fossil record as in our own bodies.

5 Answers to exercises

1, page 4. The most powerful conclusion that we have reached so far is that NN are complex entities. Therefore, every extrapolation must be carefully justified. In regard with our implementation of the strategy of divide and conquer over possibly the simplest toy model, complexity prevents us from making any generalization. This is a clear invitation for research.

2, page 12. Program `EvolNNGradualism.html` was run on a PC with 16 Gigabytes in RAM. The program needed some twenty minutes to run during 20000 generations with the following parameters:

```
POP_SIZE = 100,
```

```
MUT_RATE = 0.0001, //per char
REC_RATE = 0.1,
```

The task was to optimize the parameters of the NN, weights and biases of each one of the two perceptrons, in order to achieve the maximal number of matches under no mismatch.

We followed the behavior of the best of each generation during a simulation. Its score grows in rather small leaps. We can distinguish two stages. In the first, improvements are common and rapid. In the second, at generation 236, large waiting time gap appears. **Perfection, in reference to the best of each generation, was achieved by generation 1593 and it was maintained until generation 20000, when the program halted.** Some details are shown in the following table:

Generation	score	matches	mismatches	cost
0	-364	366	146	100
236	320	480	32	23700
1568	386	491	21	156900
1586	392	492	20	158700
1593	512	512	0	159400
20000	512	512	0	2000100

If one runs the experiment anew, one might reach perfection much faster than this and in other cases slower. Perfection is conserved.

8, page 16. Let us prove that the used architecture admits a valuation with hundreds of matches. The valuation stems from our results published in our Intro. We used there integers but we use real numbers here. There is no conflict because integers are also real numbers. This shows that valuations that produce matches exist but are so few that it is not guaranteed to find one of them by mere randomness even when mutation and recombination are added.

According to our Intro, perceptron P1 has the following parameters

```
The NN  has two perceptrons, Perceptron1, P1,
and Perceptron0, P0.

P1, had the following parameters:

inputs (the bitmap of 1) = [0, 1,  0,  0, 1, 0, 0, 1, 0 ]
weights = [-1, 1, -1, -1, 1,-1,-1, 1,-1];
threshold = 9;
bias = 9.8; //
```

In the gradualism considered here, our chromosomes are encoded in the following order: 8 weights and bias. Each parameter is encoded as a substring with 5 digits. The first integer encodes for the sign, + when the digit is even, -1 when it is odd. The other four integers represent the value of the parameter that can be in one of two formats 12.34 or 123.4. This is an election that can be made in the method `descipherSubstring(s)` on line 1250 of program `EvolNNGradualism4.html`. There was a threshold that was put at zero 0.

The weights of P1 are:

```
 -1    1   -1   -1    1   -1   -1    1   -1
10100001001010010100001001010010100001001010100
```

The bias for this program is found by subtracting the threshold from the bias that were reported in the Intro, because the threshold is absorbed by the bias. It is $9.8 - 9 = 0.8$. Hence the associated string is `00080`.

Concatenating this short string to the long one, we get the first semi-chromosome:

```
1010000100101001010000100101001010000100101010000080
```

The other semi-chromosome is found as follows. The next is the reported information about P0:

```
The NN  has two perceptrons, Perceptron1 and Perceptron0

 Perceptron0, P0,  had the following parameters:
inputs (the bitmap of 0) = [1, 1, 1, 1,  0, 1, 1, 1, 1 ];
weights =  [1, 1, 1, 1, -1, 1, 1, 1, 1];
threshold = 9;
Total bias =  4.8.
```

The weights of P0 and their associated strings are:

```
  1     1     1     1    -1     1     1     1     1
00100 00100 00100 00100 10100 00100 00100 00100 00100
```

The reported bias is 4.8 and the threshold 9, hence the updated bias is $4.8 - 9 = -4.2$ that is encoded as `10420`. Concatenating all, we get the sub-chromosome for P0:

```
00100001000010000100101000010000100001000010010420
```

The two sub-chromosomes associated with P1 and P0 were joined together and the final string was cloned to be used as initial population of chromosomes in our simulation of gradualism. That was done in program `EvolNNGradualism4.html` in the method `createPop()` in lines 1088-1093.

Program `EvolNNGradualism4.html` to be correct must fulfill a prediction: the given parameters must cause the NN to reach 260 matches under 0 mismatches out of 512 instances. This program must reproduce exactly these result, otherwise it is wrong. Moreover, this program is a slight variation of program `EvolNNGradualism3.html`. So, if this is wrong, the other is too. And if this is right, the other too. Remember that `EvolNNGradualism3.html` unveiled a failure of our gradualism to defeat the total chaos created by the threshold.

Result: the program fulfills the expectation: it reproduced 260 matches under 0 mismatches. That is why we consider that these programs are correct and this reinforces our belief that the threshold

is the cause of all problems, i.e., of the incapability of Evolution to find a valuation with a large number of matches under 0 mismatches.

9, page 17. We already have said why we believe that program `EvolNNGradualism4.html.` is correct: this program includes a threshold at 0, and exactly reproduces the expected answer that was given in the encoding of the parameters whose values were found in our Intro. The resultant NN produces 260 matches under 0 mismatches from a total of 512 options. This means that there are 512 - 260 = 252 unknowns. That program was run with a population of 2 individuals during 0 generations, i.e., just the initial behaviour was calculated. And the expectations were fulfilled. To assess the effect of the encoding of parameters as real numbers, we run the same program during many generations to see what happens: Does Gradualism improves the mark of our Intro? To facilitate rerunning by the Reader, the change appears in program `EvolNNGradualism5.html.`

What positive results did we get?

The program was run with a population size of 100000, mutation rate = 0.0001, recombination rate = 0.1, number of generations = 100. In generation 50, the program reported the following:

```
Generation = 50    bestSeq = 10100001001010010100001901010
vWeightsP1 = -1,1,-1,-1,1.9000000000000001,-1,-1,0,-1
biasP1 = 0.8
vWeightsP0 = 1,1,1,1,-1,1,1,1,1
biasP0 = -4.2
score = 274
numbOfMatches = 274
numbOfMismatches = 0
Cost = 510000
```

This represents an improvement from 260 to 274 matches under 0 mismatches. This says that the encoding of values as real variables

pays a little, and hence, it is not despicable. Anyway, the liberation from the threshold is the most important ingredient of Gradualism because it allowed to go from 260 matches to 512 that is perfection.

10, page 17. We propose the following solution to the sign problem for the Evolution of a toy NN when a threshold at 0 is included. Firstly, use the signless encoding 123.45 that works for numbers from 0 to 999.99. To recover sign, subtract 500 of the decoded number. For example, the string 12345 encodes for the real number 123.45 that next is shifted to the negative number $123.45 - 500 = -376.55$. This was done in program `EvolNNGradualism6.html`.

What did we get?

We run the program during 100000 generations with a population size of 100, mutation rate =0.0001. This was done during a night because a dedicated CPU needed 44 sec to run 100 generations and hence more than 12 hour to run 100000.

Our gradualism in the parameters and encoding was reflected in the growing of the score that combines a reward for matches plus a chastise against mismatches. The program reported the top score and the corresponding matches and mismatches. The top number of matches reached 432 by generation 53200 and remained so until generation 100001. During the Evolutionary Process, the number of mismatches was maintained low but at the end, it turned out to be 0. The number of matches grew in jumps, that were sparse but large at the beginning, for example, $0 \to 56 \to 140 \to 152 \to 276$ in 33600 generations but then decelerated in magnitude, example $375 \to 380 \to 381 \to 384$ during approximately 8000 generations.

The present behavior with our toy NN is simply described as follows: *In the engineered Gradualism, tiny mutations produce changes in the output variables, that are relatively large but sparse at the beginning of the Evolutionary Process to next decelerate in the nearness of a local optimum.* ***The Evolution of perfection is clearly visible.*** *If the mutation rate is low, the local optimum is conserved during a long period of time.* The only reason that we have here

to speak of a local and not global optimum is that we have no idea of the identity of the global optimum. Without threshold, it is 512. But with a threshold as here, we do not know what is the maximal possible number of matches without one single mismatch.

11, page 17. We are making Applied Directed Evolution, i.e., we have a goal and we put together whatever could be needed to fill it. The difference with Natural Evolution is that it has no goal. What is the arisen problem? It is that there is no enough resources to produce something useful. Think of a ship that is placed at a random place on the ocean and left to the drift by winds and waves. Does somebody expect it to arrive to some port? That person hs too much faith. What we believe is that goalless Evolution is possible to the price of wasting almost infinite resources.

12, page 17. According to our simulations, Directed Evolution cannot be made responsible for the appearing of life and especies on Earth because allowed time is insufficient. This verdict is accompanied with this other observation: Directed Evolution produces to much garbage. We immediately deduce from this that the Evolutionary Theory is false because Natural Evolution is goalless and therefore slower and produces more garbage than directed Evolution. While this assertion is intuitively acceptable, to proof it explicitly is a magnificent challenge. At the meantime, let us clarify that in relation to the development of software by Natural Evolution, to produce immense volumes of garbage implies that, apart from common extinctions, the fossil record together with our bodies must be full of malformations and malfunctions.

6 THE BIBLIOGRAPHY

References

[1] Amoutzias GD, Nikolaidis M, Tryfonopoulou E, Chlichlia K, Markoulatos P, Oliver SG. The Remarkable Evolutionary Plasticity of Coronaviruses by Mutation and Recombination: Insights for the COVID-19 Pandemic and the Future Evolutionary Paths of SARS-CoV-2. Viruses. 2022; 14(1):78. https://doi.org/10.3390/v14010078 7

[2] CHARLES DARWIN, M.A. 1859
THE ORIGIN OF SPECIES BY MEANS OF NATURAL SELECTION, OR THE PRESERVATION OF FAVOURED RACES IN THE STRUGGLE FOR LIFE. LONDON: JOHN MURRAY, ALBEMARLE STREET.
http://darwin-online.org.uk/content/frameset?itemID=F
5

[3] Richard Dawkins 1986
The Blind Watchmaker: Why the Evidence of Evolution Reveals a Universe without Design. Norton & Company, Inc
https://terebess.hu/keletkultinfo/The_Blind_Watchmaker
6

[4] Divide and Conquer Algorithm (2024)
https://www.geeksforgeeks.org/divide-and-conquer/
4

[5] Michael Nielsen (2018)
Neural Networks and Deep Learning .Determination Press 2015.
http://neuralnetworksanddeeplearning.com/
Code samples for "Neural Networks and Deep Learning"

https://github.com/mnielsen/neural-networks-and-deep-l
12

[6] José Rodríguez (2010)
The genome is software and evolution is a software developer .
http://arxiv.org/abs/1003.0575 6

[7] José Rodríguez (2021a)
The Darwinian Evolutionary Theory at its Face Value.
https://ejristos.com/dl/DarwinAtFaceValue.pdf
7

[8] José Rodríguez (2024a)
Evolutionary Neural Networks 1. Introduction.
https://www.ejristos.com/Evol%20Neural%20Networks/EvolNNintro.pdf 3

[9] José Rodríguez (2024b)
Evolutionary Neural Networks 2. Simplification.
https://ejristos.com/Evol%20Neural%20Networks/EvolNN2.pdf 3

[10] José Rodríguez (2024c)
Evolutionary Neural Networks 3. Divide and conquer.
https://www.ejristos.com/Evol%20Neural%20Networks/EvolNN3DivConquer.pdf 4

[11] José Rodríguez (2021d)
Falsification of the Evolutionary Theory with Flipp.
https://ejristos.com/GPfalsifiesEvo/FalsificETwithFlipp.pdf 7

[12] José Rodríguez (2022e)
Falsification of the Evolutionary Theory with MicroLisp.
https://ejristos.com/GPfalsifiesEvo/FalsififETwithMicroLisp.pdf 7

These links were active by 17/May/2025