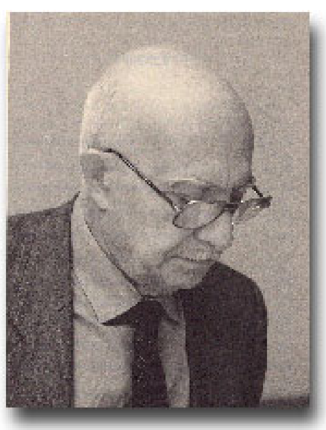
CAIANIELLO and NEURAL NETS

Paul Cull
Computer Science Dept.
Oregon State University
Corvallis, OR 97331 USA
(pc@cs.orst.edu)

1 Introduction

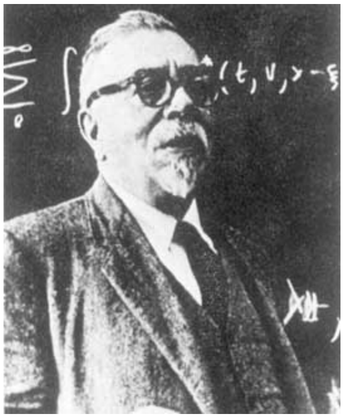


We are gathered here today to celebrate the life and scientific career of Prof. Eduardo Caianiello. It is my task to present and assess one small area of his work - neural nets.

I first met Prof. Caianiello many years ago when he visited the University of Chicago. After writing my thesis on neural nets, I subsequently worked with Prof. Caianiello on various occasions at the Laboratorio di Cibernetica and at the University of Salerno. Hence my discussion will be biased by personal recollections and hence should count more as scientific heritage rather than as strict history.

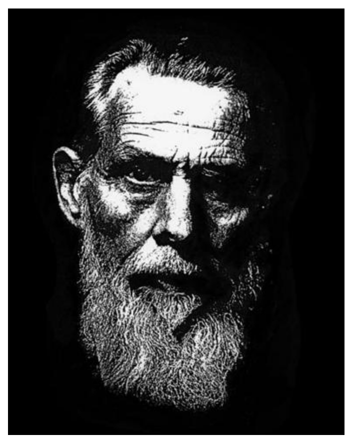
2 Pre-History

As most of you know, Caianiello's training and initial scientific work was in physics. But in the post-war period there were forces driving science in new directions. In particular, Norbert Wiener had proposed a new science - Cybernetics [22], which he called the study of control and communication in the animal and the machine. Physics dealt with matter and energy and their various transformations. Cybernetics instead would deal with information. Wiener was not calling for a rejection of physics but for an emphasis on information with energy taking a less important role. For example, he pointed out that the study of biology in terms of energy transformation was all well and good, but such study ignored the central questions of living organisms. How could a mass of atoms be coordinated to behave like a living organism? How could an organism make its way and continue to exist in some environment? These questions could not be answered by looking mainly at energy. The study of information particularly within a biological context was needed.



A start on such a study was made in the 1940s by Warren McCulloch and others. His idea may be paraphrased in syllogistic form as:

> The brain carries out logical thinking.
> Logic describes logical thinking.
> Therefore, the brain's function can be described by logic.



To these ends McCulloch and Pitts [21] created a logical calculus for brains. Their model based on logic and the then available physiology was called a neural net. The neurons in their model could be represented as:

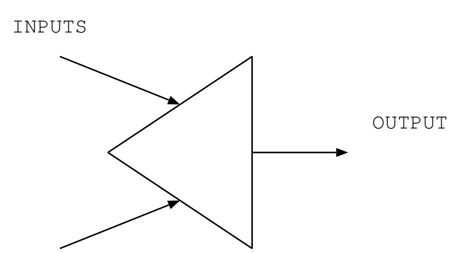


Figure 1: A neuron.

with the following assumptions:

- 0-1 Law: Inputs and outputs have only two possible values

  OFF = FALSE = 0

  ON = TRUE = 1

- Time Delay: The output at time t is a function of the inputs at time $t-1$.

- Weights on inputs:

  $a_j > 0$ Input j being ON tends to turn the neuron ON.

  $a_j < 0$ Input j being OFF tends to turn the neuron OFF.

- Total input: $Tolal \ = \ \sum_j a_j \ input_j$

- Threshold: h

- Output:

  if $Total - h > 0$ output is ON.

  if $Total - h < 0$ output is OFF.

Such neurons are called linear threshold neurons. (McCulloch and Pitts actually considered the linear threshold neurons as one of several possible models.) Neural nets could be made by connecting a set of these neurons.

In vector-matrix form a linear threshold net can be described as

$$X_{t+1} = 1(AX_t + BY_t - h)$$

where X_t and X_{t+1} are the vectors of the states of the neurons at times t and $t+1$, Y_t is the vector of external inputs, h is the vector of thresholds, A and B are matrices containing the weights for the connections between neurons and between inputs and neurons, 1( ) is the Heaviside nonlinearity extended to vectors, so that for each vector components

$$1(x) = \begin{cases} 1 \text{ if } x > 0 \\ 0 \text{ if } x \leq 0 \end{cases}$$

A neural net has a *finite* number of states with two possible states for each neuron. The behavior of a net can be described by the mappings among these states caused by the input vectors.

McCulloch and Pitts presented a number of small nets and described their behaviors. But there was (and still is) a major problem in describing the behavior of large nets. A neural net with n neurons has 2^n states, so the description of the behavior may be MUCH BIGGER than the description of the net.

3 Caianiello's Program

The sixties were a time of foment and upheaval in society and in science. One of the signs of scientific change was the introduction of new journals. The new Journal of Theoretical Biology appeared with the new decade. In the first volume of the journal Caianiello published his "An Outline of a Theory of Thought-Processes and Thinking Machines" [4] which laid out his research program on neural nets. As we will see this program led to many advances in the past forty years and continues to be influential.

The major innovation in the paper was to break the study of neural nets into two parts - *dynamics* and *learning*. This reductionistic idea may have come from Caianiello's background in physics where breaking systems apart and studying the parts in isolation have proved remarkably fruitful. As in physics, dynamics studies how the state of a system changes over time. Caianiello's insight was to assume that the states of the neurons changed so quickly that the parameters of the net could be assumed to be constant. Also, in most cases, the state changes would be faster than changes in the environment, and thus inputs could also be assumed to be constants.

In contrast to this fast dynamics, Caianiello assumed that neural nets could change and adapt to their environment, but that these adaptive changes would be slow compared to the fast dynamics. In the model these slow changes would occur in the parameters - the interconnection weights and the thresholds.

This dichotomy in time scales could reasonably describe how an organism could react to current inputs but gradually adapt and become more efficient. We will discuss this two-pronged program in the next two sections on *dynamics* and *learning*.

J. Theoret. Biol. (1961) **2**, 204–235.

Outline of a Theory of Thought-Processes and Thinking Machines

E. R. Caianiello

Istituto di Fisica Teorica dell'Università di Napoli
(Sottosezione di Napoli dell'Istituto Nazionale di Fisica Nucleare)

(*Received 9 December* 1960)

Thought-processes and certain typical mental phenomena are schematized into exact mathematical definitions, in terms of a theory which, with the assumption that learning is a relatively slow process, reduces to two sets of equations: "neuronic equations", with fixed coefficients, which determine the instantaneous behavior, "mnemonic equations", which determine the long-term behavior of a "model of the brain" or "thinking machine". A qualitative but rigorous discussion shows that this machine exhibits, as a necessary consequence of the theory, many properties that are typical of the living brain: including need to "sleep", ability spontaneously to form new ideas (patterns) which associate old ones, self-organization towards more reliable operation, and many others. Future works will deal with the quantitative solution of these equations and with concrete problems of construction—things that appear reasonably feasible. With a transposition of names, this theory could be applied to many sorts of social or, more generally, "collective" problems.

1. Introduction

A. LEVELS OF APPROACH

Attempts at a quantitative understanding and analysis of thought-processes, with or without the explicit aim of devising machines that should reproduce functions typical of the living nervous system, date as far back as Ramon Lull's syllogistic wheels. They have become a recognized and major part of scientific investigation since N. Wiener's celebrated enunciation of the principles of Cybernetics; herein lies indeed clearly, much more than in specialized studies of circuitry or of information theory, the heart and scope of this new science, which aims at synthesis as well as analysis.

The investigation of the mechanism of thought has been undertaken with a variety of methods, ranging e.g. from the study of systems that should mechanize the operations of Aristotelian logic without any requirement of similarity to living structures, to the faithful electronic reproduction of populations of hundreds or thousands of neurons. We shall benefit

204

4 Dynamics

The dynamics of neural nets may be quite complicated. There are many ways in which a net may respond to inputs. To simplify matters, Caianiello assumed that a net responded so quickly that the external inputs could be assumed to be constant. Then the inputs could be absorbed into the parameters leading to an *autonomous* neural net equation:

$$X_{t+1} = 1(AX_t - h)$$

A state of such an autonomous system uniquely determines the next state of the system. So the behavior can be described as a finite directed graph in which each node represents a state and because of the assumed autonomy, there is exactly one arrow out of each node. This limits the types of behavior to a few possibilities. A state may be a *fixed point*, that is, the only out arrow returns directly to the state.

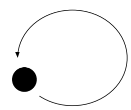


Figure 2: A fixed point.

More generally, a state may be *cyclic*, that is, following the arrows from a cyclic state, one will eventually return to that state.

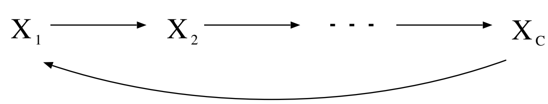


Figure 3: A cycle.

If a state is not cyclic, it is a *transient* state, that is, following the arrows from a transient state, one will never return to that same state.

The overall behavior of an autonomous net would consist of a number of *basins of attraction*, where a basin is a cycle together with all the transient states which eventually lead to that cycle. A basin could look like:

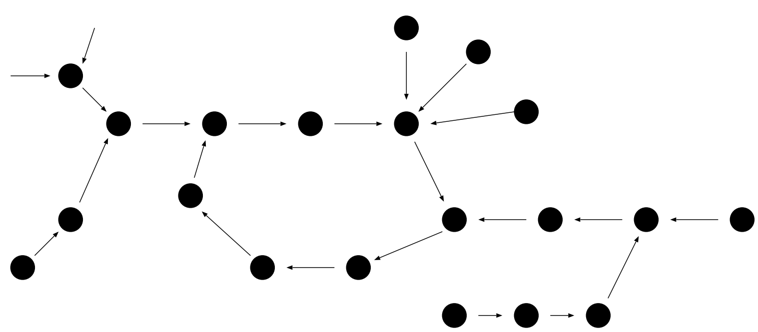


Figure 4: A basin.

4.1 Simple Examples

The following are two simple neural nets and the diagrams of their dynamics

$$X_{t+1} = 1\left(\begin{bmatrix} 5 & -4 \\ 7 & 3 \end{bmatrix} X_t - \begin{bmatrix} 5 \\ 5 \end{bmatrix}\right)$$

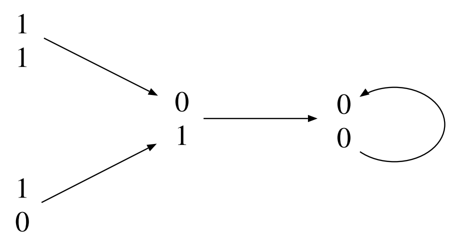


Figure 5: Example 1.

Here there is a single basin of attraction and all states lead to the same fixed point.

$$X_{t+1} = 1\left(\begin{bmatrix} -4 & 6 \\ 7 & 3 \end{bmatrix} X_t - \begin{bmatrix} 5 \\ 5 \end{bmatrix}\right)$$

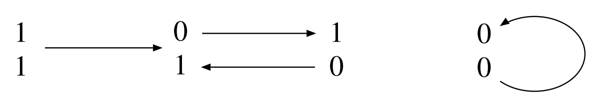


Figure 6: Example 2.

Here there are two basins. The starting state of the net will determine which basin the net is in. In one basin, the net stays fixed at $\begin{smallmatrix}0\\0\end{smallmatrix}$ while in the other basin the net will eventually oscillate with period 2.

4.2 Questions About Dynamics

Obviously, given any net one can work out its dynamics diagram. But one might also hope to be able to answer more quickly such questions as:

1. How many fixed points are there?
2. What periods occur in cycles in the net dynamics?
3. How many basins of attraction are there?
4. What is the size of the largest basin of attraction?
5. How long is the longest chain of transient states?

Caianiello also recognized that the nets with different parameters may behave in the same way. For example, changing a parameter by a small amount will probably leave the dynamics unchanged. So Caianiello said that two nets were *equivalent* iff

$$\forall X \in \{0,1\}^n \quad 1(A_1 X - h_1) = 1(A_2 X - h_2).$$

While such equivalent nets have exactly the same dynamics diagram, other nets can have the same dynamics diagram except for the name of the states. So Caianiello defined two nets as having *equivalent dynamics* iff there is an invertible function f so that

$$\forall X \in \{0,1\}^n \quad f(1(A_1 X - h_1)) = 1(A_2\, f(X) - h_2).$$

5 Linearity And Neural Nets

Caianiello and co-workers {see the references } set about studying the dynamics of autonomous neural nets. Since linear methods had proved so successful in physics, Caianiello decided to rearrange the neural net equations so that the role of the connection matrix became more prominent.

The equation

$$X_{t+1} = 1(AX_t - h)$$

is replaced by

$$Y_{t+1} = \frac{1}{2} A\ sgn(Y_t).$$

where

$$Y_t = AX_t - h$$

$$sgn(Z) = \begin{cases} 1 \ \text{ if } Z > 0 \\ 0 \ \text{ if } Z = 0 \\ -1 \ \text{ if } Z < 0 \end{cases}$$

(Notice that the Heaviside nonlinearity can be written in terms of the *sgn* nonlinearity $1(Z) = \frac{1}{2}(1 + sgn(Z))$.)

With the assumption that *no* component of Y_t is ever 0, and the assumption that the threshold vector h satisfies

$$h = \frac{1}{2} A\vec{1},$$

Caianiello could rewrite neural nets as *normal* nets,

$$Y_{t+1} = \frac{1}{2} A\ sgn(Y_t).$$

This form has only +1 and -1 as components of the Y vector and so in some sense this is more symmetric that the more usual 0,1 vector representations. But the point was to display the matrix A and to exploit the properties of this matrix.

One of the most important properties of a matrix is its *rank*. This rank equals the number of linearly independent columns in A. Vectors $v_1, v_2, \ldots, v_k$ are *linearly independent* iff $\sum c_i v_i = 0$ implies $c_1 = c_2 = \cdots = 0$.

Caianiello and co-workers derived several results from the rank of A:

- If A has rank 1, then for each Y either

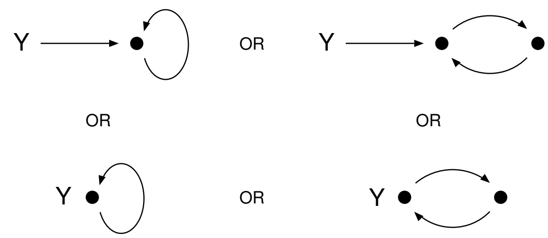


Figure 7: Dynamics.

  that is the dynamics of a rank 1 net is extremely simple.

- If an n neuron net has A with rank k, then the number of cyclic states is

$$\leq \; 2^n - 2^{n-k+1} + 2.$$

- In particular, if $k = 1$, there are at most 2 cyclic states, while if $k = n$, there may be 2^n cyclic states, i.e. every state may be cyclic.

In dynamics in physics, various quantities, like total energy, do not change even as the physical system changes states. These conserved quantities are called *constants of the motion*. For a neural net with n neurons and $rank(A) = k$, there are $n-k$ constants, $c_1, c_2, \ldots, c_{n-k}$ and $n-k$ vectors $v_1, v_2, \ldots, v_{n-k}$, so that for all t,

$$v_i \cdot X_{t+1} = c_i$$

Further given $c_1, \ldots, c_{n-k}$ and $v_1, \ldots, v_{n-k}$, one can construct a net with these constants of motion.

6 Linearizations

Some neural nets are equivalent to *linear systems*

$$X_{t+1} = AX_t$$

where A is a 0,1 matrix and the operations are MOD 2 (e.g. $1+1=0$). For such systems the fixed points are easy to find, the number of transients is easy to compute, but the lengths of the cycles may be more difficult to compute. [15]

Are all neural nets equivalent to linear systems?
YES! If one uses a 2^n dimensional linear system, any n neuron can be represented as

$$Y_{t+1} = TY_t$$

where T is a $2^n \times 2^n$ {0,1}-matrix which represents the permutations and projections of the states. Each state is represented by a {0,1}-vector with exactly one 1. This is called the *transitional* representation. Other linearizations are possible.

Every neural net can be represented as

$$X_{t+1} = F(X_t)$$

where F is a non-linear function from $\{0,1\}^n$ to $\{0,1\}^n$. Any function from $\{0,1\}^n$ to $\{0,1\}^n$ can be represented as a polynomial in n variables in which no variable has degree greater than one. A product of such polynomials is also such a polynomial. Let

$$F(X_t) = \begin{pmatrix} f_1(X_t) \\ f_2(X_t) \\ \vdots \\ f_n(X_t) \end{pmatrix} \qquad \text{where} \qquad X_t = \begin{pmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{pmatrix}$$

then if products of variables are taken in the order

$$1,\ x_1,\ x_2,\ x_1x_2,\ x_3,\ x_1x_3,\ \ldots,\ \ x_1 \ldots x_n$$

and the products of the functions are taken in a similar order

$$1,\ f_1,\ f_2,\ f_1f_2,\ f_3,\ f_1f_3,\ \ \ldots,\ f_1 \ldots f_n$$

then

$$X_{t+1} = F(X_t)$$

can be represented as

$$\mathcal{X}_{t+1} = \mathcal{F}\mathcal{X}_t$$

where $\mathcal{X}$'s are 2^n dimensional vectors and $\mathcal{F}$ is a $2^n \times 2^n$ matrix. Further $T \simeq \mathcal{F}$ over the field with two elements GF(2). There is a self-inverse matrix H so that

$$TH = H\mathcal{F} \qquad \text{and} \qquad HTH = \mathcal{F}$$

The *trace* of a matrix is the sum of the diagonal elements of the matrix. For the transition matrix T,

$$\text{trace}(T) \ = \ tr(T) \ = \ \sum T_{ii}$$

is the number of fixed points in the dynamics of the neural net. Since trace is invariant under similarity transformations, $tr(T) = tr(\mathcal{F})$. Unfortunately, the similarity is over GF(2) so this equality is a MOD 2 equality and as such it will only allow one to discover whether the number of fixed points is odd or even.

To circumvent these MOD 2 limitations, Caianiello and co-workers sought and found a linearization over a much bigger field - the rationals. As before, they replaced the state set {0, 1} by the state set {-1, 1} and then considered functions from $\{-1,1\}^n$ to {-1, 1}. All such functions can be represented as polynomials over the rationals with rational coefficients where denominators are at most 2^n. Because of the closure of the set of polynomials under addition and multiplication, there is a rational field linearization

$$\mathcal{X}_{t+1} = \mathcal{F}\mathcal{X}_t$$

for every neural net.

Again this function matrix $\mathcal{F}$ is similar to the transition matrix T. Since this similarity is over the rationals,

$$tr(T) = tr(\mathcal{F})$$

over the rationals, and $tr(\mathcal{F})$ gives an exact count of the number of fixed points in the net's dynamics.

7 Expected Behavior

Even though it is algorithmically possible to compute the dynamic behavior of a neural net, the computational effort grows like 2^n, making such computation practically impossible even for moderate values of n. But it might be possible to average over a class of nets and say something about the expected behavior of nets in this class. The obvious hope is that such average computations could be done in reasonable amounts of time.

The effect of inter-neuron connectivity on dynamic behavior was investigated by Kauffman.[19] Using a combination of simulations and theoretical calculations, he investigated the following problem:

> Pick k and let each neuron compute a function of the states of k neurons. Assign these functions at random. How does the length of cycles depend on k?

He found that for $k \approx 2$ the cycle lengths were proportional to n the number of neurons, but for large values of k, the cycle lengths were about $2^{n/2}$, that is about the square root of the number of states.

Since taking averages over the rationals makes sense, the {-1, 1} linearization seemed ideal to address this problem.

Specifically,

$$tr(T^m) = \quad tr(\mathcal{F}^m) = \quad \begin{cases} \text{Number of states in cycles} \\ \text{whose lengths divide m.} \end{cases}$$

$$E(tr(T^m)) = \quad E(tr(\mathcal{F}^m)) = \quad \begin{cases} \text{Expected number of states in cycles} \\ \text{whose lengths divide m.} \end{cases}$$

Some of the results obtained are the following:

Theorem 1. *For any probability distribution which is symmetric*

$$\text{Prob}(f_i = 1) = \text{Prob}(f_i = -1)$$

and has functions assigned independently,

$$E(tr(T)) = 1,$$

i.e. expect one fixed point.

Theorem 2. *If the functions are assigned using the uniform distribution over functions,*

$$E(tr(T^m)) = \frac{2^n!}{2^{nm}(2^n - m)!}$$

From this latter theorem, Kauffman's results for $k = n$ nets could be derived. This linearization technique can probably be used to answer some other questions about expected behavior of certain classes of neural nets.

8 Learning

The second prong of Caianiello's program for neural nets was the study of learning in such nets. Specifically, Caianiello suggested a *mnemonic equation* which described how nets could learn by association. The basic idea is that the coupling from neuron j to neuron i increases if neuron i is ON at time t and neuron j was ON at time $t-1$. Caianiello was able to give examples of learning using his mnemonic equation.

Caianiello's emphasis on association was heavily influenced by Valentino Braitenberg. Braitenberg's delightful book "Vehicles" [3] is a must-read for all those interested in the potential of learning in neural nets.

Learning has become the major focus of workers in neural nets. For example, conferences and journals like COLT, Machine Learning, NIPS, INNS, etc. attest to the great activity in this area. (Dynamics of neural nets are almost never mentioned in these venues.) A myriad of learning algorithms have now been proposed and studied. The breakthrough which produced the flowering was the replacement of the "hard" nonlinearity with a "soft" nonlinearity, that is,

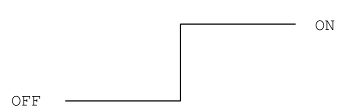


Figure 8: Heaviside function.

was replaced by

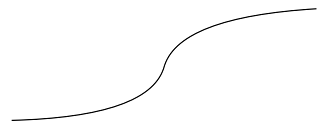


Figure 9: Sigmoidal function.

The sigmoidal nonlinearities were differentiable and this allowed algorithm designers to use calculus based optimization techniques to find "best" weights for a neural net. Many of these algorithms are not based on association, so in some sense they generalize Caianiello's ideas for neural net learning.

Active work over the last 20 years has shown that even simple feedforward networks can learn the solutions to some problems. More recent theoretical work has argued that for many problems no practical learning algorithms can exist.

9 Computational Complexity

Computational Complexity [13,17,18] was developed as a subfield of computer science in an attempt to classify which problems have reasonable algorithms and which problems could have only unreasonable algorithms. In particular, the idea of reasonable algorithms was identified with algorithms whose running time was bounded by a polynomial in the size of its input. If n was the size of the input then an algorithm with n^3 run time would be reasonable, but an algorithm with run time 2^n would be unreasonable. Various complexity classes like $\mathcal{NP}$, co-$\mathcal{NP}$, and PSPACE were defined, and it was possible to show that certain problems were the hardest problems within one of these classes. Although there are still some unsolved questions about this theory, it is generally accepted that these hardest problems will not have reasonble algorithms.

The following diagram shows these classes and indicates the positions of some problems within these classes.

Problems with reasonable algorithms are within the inner circle in this diagram. Among the harder problems for neural nets are:

- EQUI: Do two neural nets have the same behavior?
- DYNAMIC EQUI: Do two neural nets after renaming the states have the same behavior?
- FIXED POINT: Does a neural net have a fixed point in its dynamics?
- LEARN: Can a neural net of a particular type learn to solve a particular problem?

The diagram indicates that these problems are among the hardest problems for various complexity classes. Thus, it is unlikely that there are any reasonable algorithms for these problems from neural net dynamics and learning. The

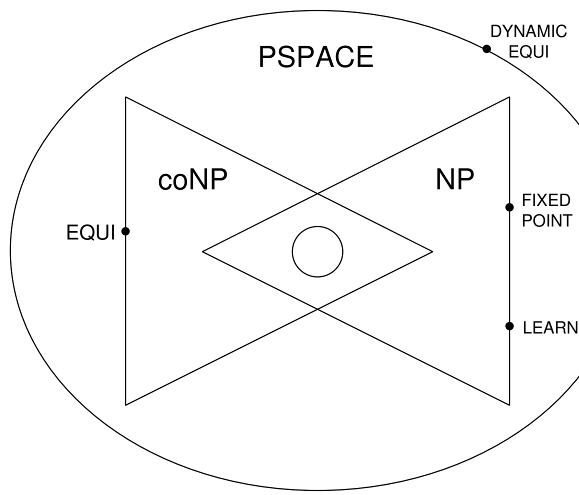


Figure 10: Complexity Classes.

challenge is to put enough meaningful restrictions on the type of neural nets so that these questions can be reasonably answered for these restricted neural networks.

In summary there are several major lessons from this line of research:

1. **Finite state models are not easy.**

One of the most attractive features about finite state models is that all questions are algorithmic, that is, it is always possible to create computer programs which are guaranteed to correctly answer the questions. Over the course of this research, it has become clear that while such programs are possible they may not be practical. Research in computational complexity has shown that various questions about neural nets are the hardest problems for various complexity classes. As such, any programs for these problems will require too much storage space and too much running time to be practical. Even answering questions for nets that are significantly smaller than vertebrate brains would require resources that would dwarf the current estimates of the size and age of the universe.

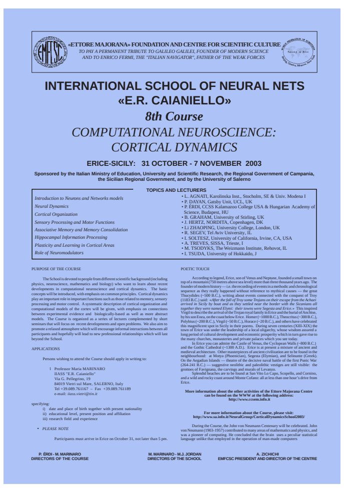
2. **Continuous differentiable models MAY be easier.**

Both dynamics and learning may be studied using continuous models which may use continuity at the neuron and subneuron level, or may model a large neural net as a continuous system. While these continuous models give up the algorithmic character of finite models, continuous models are often amenable to approximation methods. Calculating the dynamic behavior of discrete neural nets may be difficult, but using continuous models and approximation methods like "mean field" may allow an easy estimate of a net's expected behavior. Replacing strict nonlinearities with continuous approximations has given rise to learning rules which seem to behave well in some circumstances. Assumptions of continuity, differentiability, and smoothness give easier-to-understand continuous models. We do not have corresponding simplifying assumptions for discrete models.

3. **Ideas from biology still needed.**

One of the reasons for studying neural nets in the first place was that they would be models of biological systems. Conversely, the actual biological nets serve as an existence proof that there are neural nets with interesting properties. It seems that in pursuit of full mathematical generality we have ignored the constraints that biology places on real nets. For example, real nets can be large but not too large, the connections between neurons show some sort of locality, brains are organized into parts with heirarchical arrangement of these parts. Such information about real biological nets will be needed to create artificial nets that are as useful as brains.

4. **The work continues.**



One of the strongest proofs of the importance of a research program is that research in the program outlives its founder. Before his death, Caianiello established the Institute for Advanced Scientific Studies at Vietri-sul-Mare. This institute with the leadership of Prof. Marinaro has continued to host and run meetings in the field of neural nets. The latest conference occured last November. Even after his death, neural net papers with Caianiello as one of the authors continued to appear. For example, "Outline of a Linear Neural Network" by Caianiello and others appeared in 1996.[12] Caianiello's ideas continue to be referenced in current research papers and books. While Eduardo is no longer with us, the work he started is being continued by other workers. His prescient paper of 1961 is still bearing fruit 40 years later.

10 References

1. Aiello, A., Burattini, and Caianiello, E. R. (1970) Synthesis of Reverberating neural networks. *Kybernetic* **7**, 191-195.
2. Alon, N., Dewdney, A. K., and Ott, T. J. (1991) Efficient simulation of finite automata by neural nets. *J. ACM* **38**, 495-514.

3. Braitenberg, V. (1984) *Vehicles*. MIT Press, Cambridge, MA.

4. Caianiello, E. R. (1961) Outline of a theory of thought-processes and thinking machines. *J. Theoret. Biol.* **1**, 204-235.

5. Caianiello, E. R. (1966) Decision equations and reverberations. *Kybernetic* **37**, 98-100.

6. Caianiello, E. R. (1988) Problems connected with neuronic equations. In *Biomathematics and Related Computational Problems*, L. M. Ricciardi, ed., Kluwer, pp. 13-26.

7. Caianiello, E. R., and Capocelli, R. (1971) On form and language: The Procrustes algorithm for feature extraction. *Kybernetic* **8**, 223-233.

8. Caianiello, E. R., deLuca, A., and Ricciardi, L. M. (1968) Reverberations, constants of motion and general behavior. In *Neural Networks*, E. R. Caianiello, ed., Springer Verlag, pp 92-99.

9. Caianiello, E. R., and Grimson, W. E. L. (1975) Synthesis of Boolean nets and time behavior of a general mathematical neuron. *Biol. Cybernetics* **18**, 111-117.

10. Caianiello, E. R. and Grimson, W. E. L. (1976) Methods of analysis of neural nets. *Biol. Cybernetics* **22**, 1-6.

11. Cianiello, E. R. and Ventre, A. (1987) On some analytic aspects of C-calculus. In *Topics in the General Theory of Structures*. Reidel, Dordrecht, Holland.

12. Caianiello, E. R., Marinaro, M., Rampone, S., and Tagliaferri, R. (1996)Outline of a linear neural network. *Neurocomputing* **12(2-3)**, 187-201.

13. Cook, S. A. (1971) The complexity of theorem proving procedures. *Third ACM Symposium on Theory of Computing*, pp 151-158.

14. Cull, P. (1993) Dynamics of random neural nets. In *Cellular Automata and Cooperative Systems*, Kluwer, pp 111-120.

15. Elspas, B. (1959) The theory of autonomous linear sequential networks. *IRE Trans. Circuit Theory* **CT-6**, 45-60.

16. Fausett, L. (1994) *Fundamentals of Neural Networks*. Prentice Hall, Englewood Cliffs, NJ.

17. Harel, D. (1987) *Algorithmics: The Spirit of Computing*. Addison-Wesley, Reading, MA,

18. Hopcroft, J. and Ullman, J. (1979) *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA.

19. Kauffman, S. A. (1969) Metabolic stability and epigenesis in randomly connected genetic nets. *J. Theoretical Biology* **22**, 437-467.

20. Kleene, S. (1956) Representation of events in nerve nets and finite automata. In *Automata Studies*, Princeton Univ. Press, Princeton, NJ, pp 3-42.

21. McCulloch, W. S., and Pitts, W. (1943) A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5**, 115-133.

22. Wiener, N. (1948) *Cybernetics: or control and communication in the animal and the machine.*.MIT Press, Cambridge, MA.

A Little Story

Many years ago, I visited Caianiello in Naples. At that time there was a severe coin shortage. The shopkeepers had to make change by giving you a small item, because they had no coins. Among the scarce coins was the 10 Lire coin: a coin of so little value that it could be ignored. But the 10 Lire coin did have some worth. Most of the apartment buildings in Naples that had elevators required this coin to operate the elevator. Having the coin could save you a long trek upstairs.

One day, Prof. Caianiello invited us to his home to celebrate the holidays. With the invitation, he gave me a 10 Lire coin for the elevator.
I was so surprised that Caianiello had these scarce coins that I mentioned it to one of my colleagues at the lab. He replied: "The coins you put into the elevator go into a little locked box - and who do you think has the key?"

I recalled this story because of an announcement from the Vatican in March 2003 that the traditional **Keys of the Kingdom** which had hung on the statue of St. Peter in the basilica had disappeared.
I had to wonder: Who do you think has the keys?