

Networks, simulation and machine learning

Juan Segurola

Core question

How did programmable computers turn networks and models into executable systems that could simulate complex processes and learn adjustable internal representations from data?

Historical and evolutionary context

Programmable computation and information theory created two postwar abstractions: procedures could be executed by general machines, and communication could be quantified independently of physical medium. A third line treated interacting units as networks. In 1943 Warren McCulloch and Walter Pitts represented idealised neurons as logical elements. In 1953 Nicholas Metropolis, Arianna Rosenbluth, Marshall Rosenbluth, Augusta Teller and Edward Teller demonstrated a powerful Monte Carlo method on an electronic computer, showing how stochastic simulation could explore systems analytically difficult to solve. Frank Rosenblatt's 1958 perceptron made adaptive weights central to a trainable classifier. In 1986 Rumelhart, Hinton and Williams showed how back-propagation could train multilayer networks to form internal representations.

What could communities know before this transition?

Mathematics could describe differential equations, probability models and networks, and analogue devices could simulate restricted systems. PIO-082 provided programmable computation and PIO-084 measurable information. The limitation was practical scale and adaptability: many coupled systems were analytically intractable, while fixed programs required humans to specify every transformation explicitly.

What changed?

Simulation became a mode of scientific inference

The 1953 Metropolis method used random sampling and acceptance rules to explore equilibrium distributions computationally. A model could now be investigated through repeated executable experiments, creating numerical evidence where closed-form solutions were inaccessible.

Networks became trainable computational structures

McCulloch and Pitts showed that networks of simplified units could implement logical functions. Rosenblatt's 1958 perceptron added an adaptive learning rule, making performance depend on data-driven weight adjustment rather than only hand-coded logic.

Internal representations could be learned across layers

Back-propagation in 1986 made multilayer networks practically trainable for many tasks by propagating error information through differentiable connections. The important epistemic change was that some useful intermediate features no longer had to be specified in advance by the programmer.

Materials became computational infrastructure

Semiconductor devices transformed switching density, speed and reliability from the late 1940s onward. This materials thread, anticipated in PIO-074, made large digital networks and later machine-learning systems physically feasible.

Why could this capability emerge?

The transition required programmable machines, information theory, probability, numerical analysis, semiconductor electronics, larger memories, datasets and communities willing to treat numerical experiments as scientific tools. It also drew on cybernetics, neuroscience metaphors, operations research and statistical learning.

What later forms of knowledge did it enable?

Simulation became central to climate, materials, molecular, engineering and economic modelling. Machine learning enabled pattern recognition, prediction, language and vision

systems, while network models became tools for studying communication, biology and social systems.

New limits and risks

A simulation inherits the assumptions of its model, numerical method and initial conditions. Machine learning can fit correlations without causal understanding, amplify bias, leak sensitive data and produce outputs whose internal basis is difficult to interpret. Large models also concentrate computational resources and energy use. Predictive performance therefore cannot substitute for validation, provenance and accountability.

What changed in the architecture of knowledge?

The computer became an **experimental medium for executable models and adaptive representations**. Knowledge could be generated by running a formal system, comparing outputs with data and iteratively changing parameters or learned weights.

Loss, exclusion and uncertainty

Present-day narratives can project modern “AI” backward onto every earlier network model. McCulloch-Pitts neurons, perceptrons, Monte Carlo simulation and back-propagation solved different problems and emerged in different communities. Current machine learning is a layered historical synthesis, not a single invention or an inevitable path.

Dependency map

- INDIRECT ← PIO-074 Electrical networks and technological systems
- DIRECT ← PIO-082 Turing and programmable computation
- DIRECT ← PIO-084 Shannon and measurable information
- INDIRECT → PIO-091 Climate science and knowledge of a coupled planet

References

1. McCulloch, W. S. & Pitts, W. “A Logical Calculus of the Ideas Immanent in Nervous Activity.” *Bulletin of Mathematical Biophysics* 5, 115–133 (1943). DOI: 10.1007/BF02478259.
2. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. “Equation of State Calculations by Fast Computing Machines.” *Journal of Chemical Physics* 21(6), 1087–1092 (1953). DOI: 10.1063/1.1699114.
3. Rosenblatt, F. “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain.” *Psychological Review* 65(6), 386–408 (1958). DOI: 10.1037/h0042519.
4. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. “Learning Representations by Back-propagating Errors.” *Nature* 323, 533–536 (1986). DOI: 10.1038/323533a0.