Marvin L. Minsky and Seymour A. Papert

Perceptrons

An Introduction to Computational Geometry

Reissue of the 1988 Expanded Edition
with a new foreword by Léon Bottou

In memory of Frank Rosenblatt

This book was set in Photon Times Roman by The Science Press, Inc., and printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Minsky, Marvin Lee, 1927–2016
Perceptrons : An introduction to computational geometry
Bibliography: p.
Includes index.
1. Perceptrons. 2. Geometry—Data processing.
3. Parallel processing (Electronic computers).
4. Machine learning. I. Papert, Seymour. II. Title.
Q327.M55 1988 006.3 87-30990

ISBN: 978-0-262-53477-2

10 9 8 7 6 5 4 3 2 1

Context

Whereas humans can learn how to perform a task, computers must be programmed. This difference has pragmatic consequences because programming is laborious and error prone. The mere existence of such a difference is also a serious issue for the cyberneticist[1] who believes that brains and machines are both information-processing devices.

The perceptron, invented in the late 1950s,[2] was considered a paradigm shift. For the first time, a machine could be taught to perform certain tasks using examples. This surprising invention was almost immediately followed by an equally surprising theoretical result, the perceptron convergence theorem,[3,4] which states that a machine executing the perceptron algorithm can effectively produce a decision rule that is compatible with its training examples (§11.1). A youthful wave of optimism took over the research community. Although it only dealt with a very specific category of tasks, namely pattern recognition tasks, the perceptron was widely publicized as the forerunner of more general learning machines.

When the first edition of *Perceptrons* appeared almost a decade later, perceptron research had not yet produced a successful application in the real world. In fact, the most obvious application of the perceptron, computer vision, demands computing capabilities that far exceed what could be achieved with the technology of the 1960s. Meanwhile, computers—the kind one programs—were quickly showing their worth for all kinds of very tangible applications such as simulation, control, and accounting.

This reissue contains the first edition text of *Perceptrons*, with fourteen chapters numbered 0 to 13. The accompanying Prologue and Epilogue were written for the 1988 expanded edition. In chapter 0, authors Marvin Minksy and Seymour Papert clearly describe their mathematical approach and announce the theoretical results that follow. In his remarkable review of the book in 1970,[5] H. D. Block wrote:

> The conversational style and the childlike freehand sketches might mislead the casual reader into believing that this book makes light reading. … The reader who tries to provide his own proofs will, I believe, soon come to appreciate the mathematical virtuosity of the authors.

Minsky and Papert also use this conversational style to stress how much they believe that a rigorous mathematical analysis of the perceptron is overdue (§0.3). They argue that the only scientific way to know whether a perceptron performs a specific task or not is to prove it mathematically (§13.5). Not only does such a proof provide an answer for the

specific task, but it also provides a framework for understanding the performance of the perceptron in general. Accordingly, Minsky and Papert's analysis focuses on characterizing the computational abilities of perceptrons for specific tasks, such as parity or connectedness, whose geometrical interpretation helps in both establishing the proofs and appreciating their consequences. Their rigorous work and brilliant technique does not make the perceptron look very good …

Following the publication of the book, research on perceptron-style learning machines remained unfashionable until the mid-1980s. Funding was no longer forthcoming. Compelled to change fields, perceptron researchers applied their unfashionable ideas wherever they went. This exodus helped spur the development of adaptive signal processing,[6] which, ironically, was one of the earliest and most spectacular applications of learning techniques. Meanwhile, somehow protected from these events by the iron curtain, V. N. Vapnik and A. Ya. Chervonenkis[7] developed the general theory of statistical learning machines.

The pendulum swung back toward learning machines in the mid-1980s with the work of the PDP Research Group[8] and the early successes of the multilayer perceptron back-propagation algorithm.[9] With this revival came the question of whether *Perceptrons* interrupted a promising line of research by overplaying the consequences of its theoretical analysis. The most direct answer can be found in Papert's 1988 *Daedalus* article:[10]

> This story seems to call for a plea of guilty or innocent: Did Minsky and I try to kill connectionism, and how do we feel now about its resurrection? Something more complex than a plea is needed. …[P]art of our drive came … from the fact that funding and research energy were being dissipated on what still appears to me … to be misleading attempts to use connectionist methods in practical applications. But most of the motivation for *Perceptrons* came from … finding the appropriate balance between romanticism and rigor in the pursuit of artificial intelligence. (p. 4)

The authors give a slightly more combative answer in the Prologue of the expanded edition of *Perceptrons* (1988): "progress had already come to a virtual halt because of the lack of adequate basic theories." In the Epilogue, they argue that the new wave of network machines still does not address their concerns about the lack of basic theory and relies on hill-climbing procedures that are potentially marred with intractable scalability problems.

In other words, Minsky and Papert's message had not changed in the twenty years that separated the expanded edition of *Perceptrons* from

the first edition. Without an adequate basic theory, they believed that attempting to use connectionist learning machines in practical applications was futile. Far from trying to kill this research, they conceived *Perceptrons* as a first step toward correcting an already fatal flaw.

Aftermath

Three decades later, machine learning is a thriving research field.[11] The effectiveness of deep neural networks for practical applications such as speech recognition and computer vision is undeniable. Cell phones have gained the capability to recognize speech and sometimes understand what we mean.[12] Automobiles are increasingly capable of driving themselves.[13] Pundits write about the imminence of artificial intelligence. Although our theoretical understanding of learning machines has clearly progressed, these applications often rely on deep neural networks for which theory still offers very little guidance. Does this contradict Minsky and Papert's argument about the necessity of an adequate basic theory? Can we sustain such progress using solely intuition and experimentation? What is the role of mathematics in these developments?

Programming a computer is by essence a very mathematical exercise. The specification of a task lists the mathematical properties that one wishes to see satisfied. The program is correct when its completion ensures that these properties are satisfied. Programs can invoke subprograms in the same way that the proof of a theorem can invoke more elementary theorems. If the subprograms are known to be correct, the designer of the main program can ignore their details and reason instead with the specifications of the subtasks. This mathematical property of programming means that, in practice, one can organize teams of programmers and clearly define their individual responsibilities. Therefore, computers excel in all domains whose essential tasks satisfy two criteria: (a) they come with clear mathematical specifications, and (b) their economic importance is sufficient to pay for an adequate team of programmers. This is true of accounting, for instance, and communication protocols, computer-aided design, and so on.

There are, however, many economically important domains that do not satisfy the first criterion. For example, in visual pattern recognition, we can determine whether a connectedness recognition program is correct using the mathematical definition of the set of images representing a connected shape (§9.2), but we cannot establish that a program that recognizes cat images is correct in the same way because we lack a mathematical description of the set of images representing a cat.

The traditional way to address the absence of a mathematical specification consists of heuristically constructing one. When Minsky and Papert argue that the connectedness predicate is an obvious subtask for problems such as recognizing chairs, tables, or people (§13.3), they in fact claim that one can provide a heuristic specification by judiciously assembling well-defined geometrical predicates such as connectedness. One can then design programs that target the heuristic specification (§13.4). Of course, this approach introduces a logical gap: although we can conceivably prove that a program fulfills the heuristic specification, this does not guarantee that the program can perform the task of interest. In the case of computer vision, scientists have devised many ingenious ways to leverage physical and geometrical insights about the nature of images.[14] However, absent a mathematical definition of what makes a cat look like a cat, the logical gap remains. Almost a proof is no proof.

The alternative way to address the absence of a mathematical specification consists of replacing the missing specification with a large collection of training images and relying on a learning machine, such as the perceptron, to construct an appropriate predicate. Instead of leveraging human ingenuity to create a heuristic specification, the idea here is to leverage the training data and the computational power available to process it. Therefore, the reach and the performance of the learning approach increase whenever more data and more computing power become available. For instance, in the case of visual object recognition, it is generally agreed that the tipping point was reached in 2012 when a convolutional neural network decisively outperformed all competing approaches.[15] This neural network was merely a very large variant of the convolutional networks introduced two decades before.[16] The key factors were the availability of a large image dataset[17] and the emergence of GPU (graphics processing unit) computing.

When they wrote *Perceptrons*, Minsky and Papert clearly did not anticipate that the mere passage of time would eventually tilt the balance in favor of learning systems. Maybe they simply placed too much trust in human ingenuity, expecting instead that these pesky pattern recognition problems would be solved long before reaching the tipping point. Nevertheless, this general trend possibly explains why learning machines have enjoyed practical successes despite the weaknesses of basic theory.

A closer analysis reveals finer trends. We have already seen that the use of heuristic specifications undermines the theoretical basis of programming. The lack of mathematical specification affects the theoretical basis

of learning machines in a slightly different way. Theoretical questions about learning machines roughly belong to two categories: can the learning machine represent the predicate of interest, and, assuming this is the case, can the learning algorithm approach the predicate of interest when the number of training examples increases? Because we cannot rely on a mathematical definition of the predicate of interest, these results must target a much greater family of predicates, and often all of them. For instance, when Minsky and Papert prove that no finite-order perceptron can represent the connectedness predicate (§5.2), they state that a finite-order perceptron cannot represent all predicates. Such a result does not necessarily mean that the learning system cannot represent the predicate that tells whether the image represents a cat. In fact, a human can easily recognize a cat but has trouble visually determining which of the two shapes shown on the cover of *Perceptrons* is connected.

We now know of learning systems for which both categories of theoretical questions have positive answers. For instance, a support vector machine using a universal kernel can represent and approach *any* predicate with arbitrary precision when the size of the training set increases.[18] Such a guarantee is stronger than anything offered by the heuristic specification approach. Many researchers in the 2000s were therefore confident that such principled learning systems would quickly dominate the practical applications of pattern recognition, only to be reminded in the 2010s that a jack of all trades is master of none. Although deep learning systems[19] do not offer such solid guarantees, they are responsible for a string of spectacular successes in economically important applications such as speech recognition, computer vision, and machine translation.

The current status of deep learning systems can be compared to the age of steam that marked the beginning of the industrial age. The steam engine began transforming the world[20] long before the formulation of the laws of thermodynamics.[21] The massive technological advances that have shaped our modern world, however, would have been unthinkable without thermodynamics. In that sense, Minsky and Papert's message remains very relevant.

Léon Bottou
February 16, 2017

Stephenson's Rocket, *Mechanics Magazine*, 1829.
Source: Wikimedia Commons user Duncharris~commonswiki.

Notes

1. N. Wiener, *Cybernetics: Or Control and Communication in the Animal and the Machine* (Cambridge, Mass.: MIT Press, 1948).

2. F. Rosenblatt, "The Perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, Vol. 65, No. 6 (1957), pp. 386–408.

3. H. D. Block, "The Perceptron: A model for brain functioning," *Reviews of Modern Physics*, Vol. 34, No. 1 (1962), pp. 123–135.

4. A. B. J. Novikoff, "On convergence proofs for Perceptrons," *Mathematical Theory of Automata: Proceedings of the Symposium on Mathematical Theory of Automata*, New York, April 24, 25, 26, 1962, ed. Jerome Fox, Vol. 12 (Brooklyn, N.Y.: Polytechnic Press, 1962), pp. 615–622.

5. H. D. Block, "A review of 'Perceptrons: An Introduction to Computational Geometry,'" *Information and Control*, Vol. 17 (1970), pp. 501–522.

6. B. Widrow and S. D.Stearns, *Adaptive Signal Processing* (Englewood Cliffs, N.J.: Prentice Hall, 1985).

7. V. N. Vapnik and A. Ya. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Proceedings of the USSR Academy of Sciences*, Vol. 181, No. 4 (1968), pp. 781–783. Translated by the American Mathematical Society as *Soviet Mathematics*, Vol. 9 (1968), pp. 915–918.

8. D. E. Rumelhart, J. L. McClelland, and the PDP Group, *Parallel Distributed Processing* (Cambridge, Mass.: MIT Press, 1986).

9. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representation by error propagation," in *Parallel Distributed Processing I—Explorations in the Microstructure of Cognition*, ed. David E. Rumelhart, James L. McClelland, and PDP Research Group (Cambridge, Mass.: MIT Press, 1986), pp. 318–362.

10. S. Papert, "One AI or many?," *Daedalus*, Vol. 117 (1988), pp. 1–14.

11. See *Advances in Neural Information Processing Systems,* Vols. 0–29 (1987–2016).

12. See "Finding a voice," *The Economist*, Technology Quarterly, http://www.economist.com/technology-quarterly/2017-05-01/language.

13. See Guilbert Gates, Kevin Granville, John Markoff, Karl Russell, and Anjali Singhvi, "When cars drive themselves," *The New York Times*, last modified March 13, 2017, https://www.nytimes.com/interactive/2016/12/14/technology/how-self-driving-cars-work.html.

14. D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, 2nd ed. (Harlow, Essex, UK: Pearson, 2011).

15. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information*

Processing Systems 25 (NIPS 2012), ed. F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Red Hook, N.Y.: Curran Associates Inc., 2012).

16. Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," *Advances in Neural Information Processing Systems 2 (NIPS 1989)*, ed. David S. Touretsky, (San Mateo, Calif.: Morgan-Kaufmann, 1989), pp. 396–404.

17. See the image database at http://www.image-net.org.

18. I. Steinwart, "Support vector machines are universally consistent," *Journal of Complexity*, Vol. 18, No. 3 (2002), pp. 768–791.

19. John Markoff, "Scientists see promise in deep-learning programs," *The New York Times*, November 23, 2012, http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html.

20. See *Stephenson's Rocket*, Science Museum, London (1829), at the end of this foreword.

21. R. Clausius, *The Mechanical Theory of Heat—with Its Applications to the Steam Engine and to Physical Properties of Bodies*, trans. Thomas Hirst (London: John van Voorst, 1867). Originally published as *Die Mechanische Wärmetheorie* (Braunschweig: Friedrich Vieweg und Sohn, 1864). The English translation contains nine memoirs published 1850–1865.

Computational geometry is a branch of computer science devoted to the study of algorithms which can be stated in terms of geometry. Some purely geometrical problems arise out of the study of computational geometric algorithms, and such problems are also considered to be part of computational geometry. While modern computational geometry is a recent development, it is one of the oldest fields of computing with a history stretching back to antiquity. This is just a slight introduction for all of you! Geometry is a branch of mathematics concerned with questions of shape, size, relative position of figures, and the properties of space. It is a branch of computer science devoted to the study of algorithms which can be stated in terms of geometry. So where is computational geometry used?Â If we look at the problems involving computational geometry, they are interesting theoretically and often involve a lot of proofs, intuitions, corollaries etc. A lot of people have the common notion that knowing a couple of problems involving geometry and not understanding how things are happening will be enough - for a while, it might just be enough, but in the long run, you might start falling short of things. to. Computational. Geometry. -M. Minsky.Â p align="center"> Abstract Computer vision is an image processing by a computer to obtain information from image captured through the camera generally used in real-time application. This paper reports on the results of research conducted on computer vision system designed to be able to recognize the image number (0-9) and mathematical operators (addition (+) and subtraction (-)) in a card number figures. Computer vision system designed in this study consists of a camera on the android phone that used to captured images on the card number and the computer that has artificial neural network perc Introduction to Computational Geometry Hackson Agenda â€¢ â€¢ â€¢. Dot, Line and Plane Cartesian Coordinate System Straight Line and Segment Distance â€“ how to measure? Cartesian Coordinate Geometry Vector Geometry Intersections Polygons To begin with DOT, LINE AND PLANE Geometry â€¢ ``Geometry (Greek Î³ÎµÏ‰Î¼ÎµÏ„Ï±; geo = earth, metria = measure) is a part of mathematics concerned with questions of size, shape, and relative position of figures and with properties of space.â€™â€™ â€“ Wikipedia â€¢ What to deal with? â€¢ â€¢ â€¢ â€¢ Lines Polygons Planes Objects (in N dimensional!) It clearly demonstrates that computational geometry in the plane is now a fairly well-understood branch of computer science and mathematics. It also points the way to the solution of the more challenging problems in dimensions higher than two." #Mathematical Reviews#1 " This remarkable book is a comprehensive and systematic study on research results obtained especially in the last ten years.Â 2.1 Introduction to Geometric Searching 2.2 Point-Location Problems 2.2.1 General considerations. Simple cases 2.2.2 Location of a point in a planar subdivision 2.2.2.1 The slab method. 36 41 41 45 45.