

The Profession of IT Learning Machine Learning

A discussion of the rapidly evolving realm of machine learning.

MACHINE LEARNING HAS evolved from an out-of-favor subdiscipline of computer science and artificial intelligence (AI) to a leading-edge frontier of research in both AI and computer systems architecture. Over the past decade investments in both hardware and software for machine learning have risen at an exponential rate matched only by similar investments in blockchain technology. This column is a technology check for professionals in a Q&A format on how this field has evolved and what big questions it faces.

Q: The modern surge in AI is powered by neural networks. When did the neural network field start? What was the first implementation?

A. The early 1940s was a time of increasing attention to automatic computers. At the time, a “computer” was a professional job title for humans and computation was seen as a human intelligent activity. Some believed that the logical computations of the brain were made possible by the neuronal structure of the brain.

In 1943 Warren McCulloch and Walter Pitts wrote a famous proposal to build computers whose components resembled neurons.⁴ Each neuron received inputs from many others and delivered its outputs to many others. Inputs had weights and when the weighted input sum exceeded a threshold the neuron switched from the 0 to the 1 state. They wrote: “Because of the ‘all-or-none’ character of nervous activity, neural events and the relations among them can be treated by means of propositional

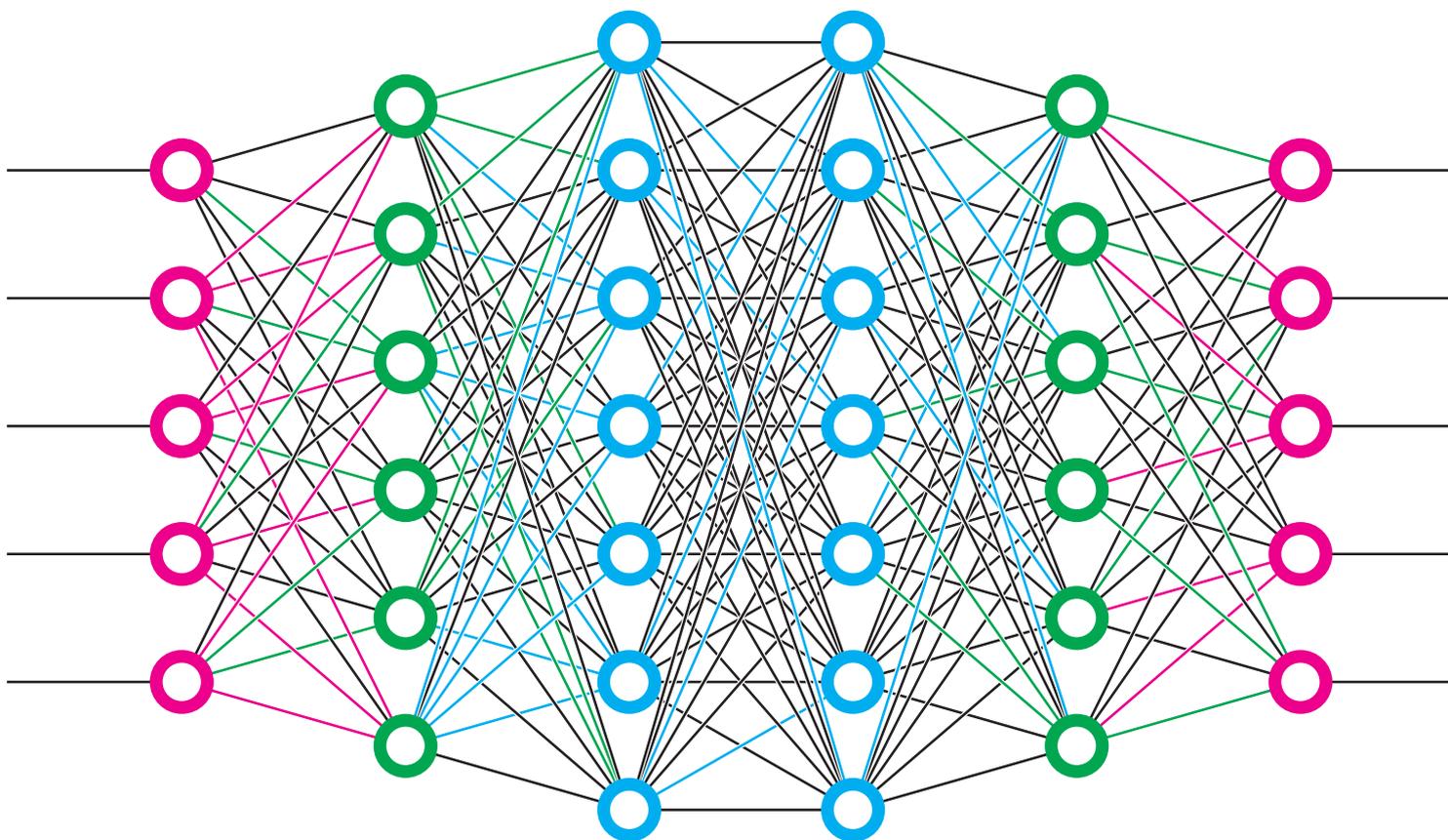
It takes an enormous amount of computation to train a large network on a large training set.

logic.” It was logical that a machine organized like the brain would be good with logic! McCulloch and Pitts established the foundation for future artificial neural networks.

In 1957, Frank Rosenblatt demonstrated the first artificial neural network machine for the U.S. Navy. He called it the Perceptron. It was a single-layer machine as illustrated schematically in the accompanying figure using photocells as input receptors organized as two-dimensional array. The Perceptron was able to recognize handwritten digits 0 through 9. The figure also outlines a genealogy of the neural network descendants of the Perceptron; we provide it for information, but we will not discuss all its branches here.

Q: Neural networks are good for mapping input patterns into output patterns. What does this mean?

A pattern is a very long sequence of bits, for example, the megabits making up an image or gigabits representing a person’s viewing history of films. A recognizer or classifier network maps a pattern into another that has meaning to humans. A recognizer network can,



for example, take the bitmap from a camera shown the digit “9” and output the ASCII code for “9”. A recommender network maps a pattern into a string representing an action the human might decide to take next. Netflix has developed a neural network that takes your entire film viewing history along with all the ratings you and others gave those items and returns a recommendation of a new film that you would probably rate highly.

Q: How do you program a neural network?

You don’t. You teach it to learn how to do the function you want. Suppose you want to teach a network a function F that maps X patterns into Y patterns. You gather a large number of samples (X,Y) , called the training set. For each sample you use a training algorithm to adjust the connection weights inside the network so that the network outputs Y when given X at its input. There are so many neurons, connections, and possible weights that the training algorithm can successfully embed a large number of pairs (X,Y) into the network. It takes

an enormous amount of computation to train a large network on a large training set. We now have the hardware power and training algorithms to do this. The trained network will implement all the trained (X,Y) combinations very reliably.

Once the network is trained, it can compute its outputs very rapidly. It has a fixed function until its training is updated.

We want to use our networks to compute not only trained maps, but untrained ones as well. That means to compute $Y=F(X)$ for a new pattern X not in the training set. An important question is how much trust can be put in the responses to untrained input patterns.

If we keep track of the new (untrained) inputs and their correct outputs, we can run the training algorithm again with the additional training pairs. The process of training is called learning, and of retraining reinforcement learning.

The network does not learn on its own. It depends on the training algorithm, which is in effect an automatic programmer.

Q: In 1969, Marvin Minsky and Seymour Papert published a book showing that Perceptrons could not recognize important patterns.⁵ What effect did that have on the field?

The entire field faltered following publication of the Minsky-Papert book. They proved that single-layered perceptrons would only work as classifiers when the data was “linearly separable”—meaning that the multidimensional data space could be separated into regions bounded by hyperplanes. Data not clustered in this way could not be recognized. This finding caused interest in perceptrons to die off, until researchers discovered that adding layers and feedback to the neural network overcame the problem.¹⁻³ The multi-layered perceptron (MLP) can reliably classify patterns of data in nonlinear spaces. More layers mean more accuracy and, of course, more computation. The modern term “deep learning” acknowledges the many-layers depth of a neural network.

An open problem for research today is finding the smallest number of neurons and layers to implement a given function.

acm

Advertise with ACM!

Reach the innovators and thought leaders working at the cutting edge of computing and information technology through ACM's magazines, websites and newsletters.



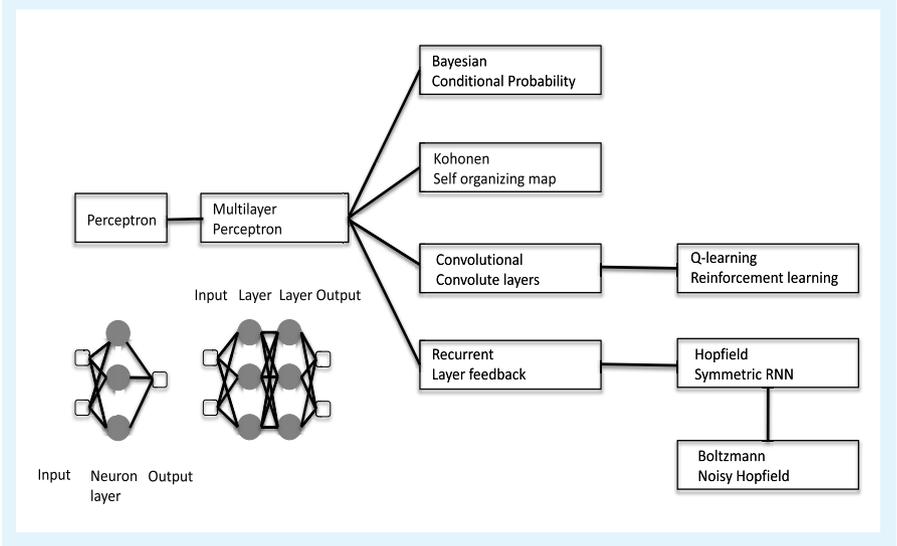
Request a media kit with specifications and pricing:

Ilia Rodriguez
+1 212-626-0686
acmm mediasales@acm.org

acm

media

Figure 1. An abbreviated taxonomy of the evolution of neural networks shows a progression from simple one-layer Perceptron to multilayered Perceptron (MLP), convolutional and recurrent networks with memory and adaptable reinforcement learning algorithms.



Back in favor because of the MLP breakthrough, neural networks advanced rapidly. We have gone well beyond recognizing numbers and handwriting, to networks that recognize and label faces in photographs. New methods have since been added that allow recognition in video moving images; see the figure for some of the keywords.

Q: What propelled the advances?

Two things. The abundance of data, especially from online social networks like Twitter and Facebook, large-scale sensor networks such as smartphones giving positional data for traffic maps, or searches for correlations between previously separate large databases. The questions that could be answered if we could process that data by recognizing and recommending were a very strong motivating force.

The other big factor is the proliferation of low-cost massively parallel hardware such as the Nvidia GPU (Graphics Processing Unit) used in graphics cards. GPUs rapidly process large matrices representing the positions of objects. They are super-fast linear-algebra machines. Training a network involves computing connection matrices and using a network involves evaluations of matrix multiplications. GPUs do these things really well.

Q: These networks are now used for critical functions such as medical diagnosis or crowd surveillance to detect

possible terrorists. Some military strategists talk about using them as automatic fire-control systems. How can we trust the networks?

This is a hot question. We know that a network is quite reliable when its inputs come from its training set. But these critical systems will have inputs corresponding to new, often unanticipated situations. There are numerous examples where a network gives poor responses for untrained inputs. This is called the “fragility” problem. How can we know that the network’s response will not cause a disaster or catastrophe?

The answer is we cannot. The “programs” learned by neural networks are in effect enormous, incomprehensible matrices of weights connecting millions of neurons. There is no more hope of figuring out what these weights mean or how they will respond to a new input than in looking for a person’s motivations by scanning the brain’s connections. We have no means to “explain” why a medical network reached a particular conclusion. At this point, we can only experiment with the network to see how it performs for untrained inputs, building our trust slowly and deliberately.

Q: Computers were in the news 20 years ago for beating the grandmaster chess players, and today for beating the world’s Go master champion. Do these advances signal a time when machines

can do all human mental tasks better than today's humans can?

First, let's clarify a misconception about the IBM computer that beat chess grandmaster Garry Kasparov in 1997. It was not a neural network. It was a program to search and evaluate board positions much faster than Kasparov. In effect, the human neural network called "Kasparov" was beaten by the IBM computer using smart search algorithms. Kasparov bounced back with Advanced Chess in which humans assisted by computers played matches; the human teams often beat solo computers.

The neural network AlphaGo won four of five Go games against the world champion Le Se-dol. According to DeepMind researcher David Silver, "The most important idea in AlphaGo Zero is that it learns completely tabula rasa, that means it starts completely from a blank slate, and figures out for itself only from self-play and without any human knowledge."^a AlphaGo Zero contains four CPUs and a single neural network and software that initially know nothing about Go or any other game. It learned to play Go without supervision by simply playing against itself. The results look "alien" to humans because they are often completely new: AlphaGo creates moves that humans have not discovered in more than 2,500 years of playing Go.

We think this is a development of singular significance. The time-honored method of neural networks learning by being trained from training sets of data can in some cases be replaced by machines learning from each other without any training data.

So far, there is little threat from these networks becoming super-intelligent machines. The AlphaGo experience happened in a game with well-defined rules about allowable moves and a well-defined metric defining the winner. The game was played in a well-defined mathematical space. Presumably this training method could be extended to swarms of robots attacking and defending. But could it master a sport like basketball? Playing a violin? And what

about games the purpose of which is to continue rather than to win?

Q: Where do you see this going, next?

The 10–15 year roadmap is pretty clear. There is now much theory behind neural networks. Even much of the software is becoming off-the-shelf through open source such as Google's TensorFlow and Nvidia's CUDA tools. The next breakthrough is likely to be in hardware. Cheaper and faster hardware will pave the way for consumer-level products that fit in a smartphone or drive a car.

Hardware is already trending toward chip sets with massively parallel neural networks built in. The Von Neumann architecture, long criticized for its processor-memory bottleneck, is giving way to processing-in-memory machines where simulated neural network nodes are embedded in memory. Imagine random access memory in small blocks, each encapsulated in a node of a neural network. Such networks will perform big-data analytics, recognition, and recommending without needing the full power of a general-purpose arithmetic logic unit. Who knows what will emerge in the worldwide network of interconnected devices bearing power neural network chips? □

References

1. Cybenko, G. Approximation by superpositions of a Sigmoid function. *Math. Control Signals Systems 2*, (1989), 303–314.
2. Hopfield, J. and Tank, D. Neural computation of decisions in optimization problems. *Biological Cybernetics 52* (1985), 141–152.
3. Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural Networks 3*, 2 (1989), 359–366.
4. McCulloch, W.S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics 5*, 115 (1943); <https://doi.org/10.1007/BF02478259>
5. Minsky, M. and Papert, S. *Perceptrons*. MIT Press, 1969.

Ted G. Lewis (tedglewis@redshift.com) is an author and consultant with more than 30 books on computing and hi-tech business, a retired professor of computer science, most recently at the Naval Postgraduate School, Monterey, CA, Fortune 500 executive, and the co-founder of the Center for Homeland Defense and Security at the Naval Postgraduate School, Monterey, CA.

Peter J. Denning (pjd@nps.edu) is Distinguished Professor of Computer Science and Director of the Cebrowski Institute for information innovation at the Naval Postgraduate School in Monterey, CA, is Editor of *ACM Ubiquity*, and is a past president of ACM. The author's views expressed here are not necessarily those of his employer or the U.S. federal government.

a <https://ab.co/2ypCCnP>

Calendar of Events

December 2–3

VRCAI '18: International Conference on Virtual Reality Continuum and Its Applications in Industry, Hachioji, Japan, Sponsored: ACM/SIG, Contact: Koji Mikami, Email: mikami@stf.teu.ac.jp

December 4–7

CoNEXT '18: The 14th International Conference on Emerging Networking EXperiments and Technologies, Heraklion, Greece, Sponsored: ACM/SIG, Contact: Alberto Dainotti, Email: alberto@caida.org

December 10–14

Middleware '18: 19th International Middleware Conference, Rennes, France, Sponsored: ACM/SIG, Contact: Guillaume Pierre, Email: guillaume.pierre@irisa.fr

December 13–14

CVMP '18: European Conference on Visual Media Production, London, U.K., Sponsored: ACM/SIG, Contact: Abhijeet Ghosh, Email: abhijeetg@gmail.com

2019

January

January 14–18

AFIRM '19: ACM SIGIR/SIGKDD African Workshop on Machine Learning for Data Mining and Search, Cape Town, South Africa Co-Sponsored: ACM/SIG, Contact: Hussein Suleman, Email: hussain@cs.uct.ac.za

January 29–31

FAT* '19: Conference on Fairness, Accountability, and Transparency, Atlanta, GA, Sponsored: ACM/SIG, Contact: danah boyd, Email: danah@datasociety.net