MAY 16, 2019

WORLDQUANT®

PERSPECTIVES

Probabilistic Programming and the Art of the Possible

Today machine learning is rigid and requires massive amounts of data. But AI researchers are attempting to emulate the low-data fluidity and rich representations of human thinking. Give credit to Thomas Bayes.

By Michael Kozlov and Ashish Kulkarni

WorldQuant, LLC
1700 East Putnam Ave.
Third Floor
Old Greenwich, CT 06870
www.weareworldquant.com

IN NATURE AND IN LIFE, MORE IS NOT NECESSARILY BETTER. A gentle rain is good for the grass, but if it lasts for five days, you may get a raging flood. Rising temperatures can be pleasant, even optimal, until the heat burns, the droughts sear and the polar ice caps start to melt. A long-popular finance theory argues that companies with too much cash on hand will begin to do stupid things.

This also applies to data. We are awash in data, and it's growing at an exponential rate. It often seems as if this tidal wave of data is a necessary prerequisite for computer programs designed to learn. Today's machine learning programs require huge amounts of data to discover what toddlers can master by crawling around and playing with blocks or simply observing the strange behavior of adults. In fact, an excess of data can quickly overwhelm the ability of human brains to function properly. Humans need a lot less data than machines to generalize about and draw conclusions on matters large and small, simple and complex.

For instance, training a deep learning algorithm to recognize a category of objects — say, chairs or dogs — with great accuracy is not easy, but distinguishing Person A from Person B is even harder and requires even more data. If you teach an artificial intelligence (AI) program that a robin, an eagle and a duck are all birds, the program still may not recognize a cardinal or a peacock as one of our feathered friends. That would require feeding the algorithm tens or even hundreds of thousands of images, capturing a massive amount of variation in size, shape, profile, texture, lighting and angle.

Handling large amounts of data is cumbersome, slow and expensive. Anyone who has tried training a convolutional neural network (CNN), which analyzes visual imagery, or a generative adversarial network (GAN), which pits two neural networks against each other in a zero-sum game, knows the large computational effort and multiple passes of training data required to achieve a robust level of accuracy.

It would be much more efficient if an algorithm could develop ideas about what makes a human a human with less data and computational power, easily test them and learn from successes and failures. One answer to this challenge is probabilistic computing, an increasingly vital component of AI and a way of addressing the uncertainties inherent in so-called natural data — the kind of real-world data that humans continually process, often without conscious intent. Advances in probabilistic computing make it increasingly likely that we'll soon be able to build machines that are capable of understanding, predicting and making decisions with limited amounts of data.

> *The model learns to learn by using previous experience with related concepts to facilitate the mastery of new concepts.*

Two aspects of conceptual knowledge have eluded machine learning systems. First, for most natural categories (birds, fish, dogs, people) and artificial categories (man-made cars, boats, smartphones), people seem to be able to learn new concepts from one or a handful of examples. Standard machine learning algorithms, on the other hand, require large numbers of examples.[1] Second, humans master richer representations of the categories than machines can, even for simple concepts; that is, they understand what inherent features make a bird a bird and a rock a rock. Humans learn these inherent features without careful study or instruction and can use them for a wide range of functions. Even when objects in a category are quite diverse, humans can create new "exemplars" — generalized concepts stored in memory — parse objects into parts, relate them to similar objects and develop new abstract categories. In contrast, the best machine classifiers fall behind when they attempt these additional functions, even with deeper analysis and specialized algorithms.[2]

A central challenge is to explain two characteristics of human conceptual learning: how people learn new concepts from just a few examples and how they generate such abstract, rich and flexible representations. An even greater challenge arises when you put the two together: How can machine learning use small amounts of data and produce rich representations? If this sounds paradoxical, it is, based on most current learning theories. Nonetheless, people seem to navigate this trade-off with remarkable agility, learning rich concepts that generalize well from relatively sparse data.[3] The answer may lie with a long-dead Presbyterian minister who had a deep interest in probabilities.

THE BAYESIAN ROOTS OF PROBABILISTIC PROGRAMMING

Probabilistic, or Bayesian, programming, is a high-level software language that allows developers to define Bayesian probability models and "solve" them automatically. Thomas Bayes was an English statistician and Presbyterian minister in the 18th century who developed what is now widely known as Bayes' theorem. This theorem is a means of updating beliefs in a hypothesis based on new evidence. It is rooted in probabilities: The more often you see the sun rise in the morning, the higher the probability that it will continue to happen in the future. The Bayesian approach captures a key aspect of how humans learn in an uncertain world. And although this learning can be quite complex, involving many hypotheses and multiple probabilities, it has led to a new form of AI that is improved by learning and inference.

Bayes published his theorem during his lifetime, but 50 years after his death in 1761 his work was revived and formalized by the great French statistician Pierre-Simon Laplace. Much more recently, researchers have attempted to develop probabilistic programming along Bayesian lines. In 2015, a team from the Massachusetts Institute of Technology and New York University used probabilistic methods to teach computers to recognize written characters and objects after seeing just one example. However, their approach proved to be little more than an academic curiosity given the difficult computational challenges, including the fact that any such program has to consider many different possibilities.

More recent developments in high-performance computing and deep learning algorithms suggest that probabilistic computing is entering a new era. In the next few years, experts anticipate research to produce significant improvements in the reliability, security, serviceability and performance of AI systems, including hardware designed specifically for probabilistic computing.

There are certainly strong incentives for developing machine learning approaches that are easier to use and less data-hungry. Machine learning currently requires a large, raw dataset, which typically needs humans to manually label it, clean it and reduce "noise" — that is, meaningless or random data. The actual learning takes place inside large data centers using many computer processors churning away in parallel for hours or days. In general, the time and cost associated with this are material.

Many researchers start their adventures in machine learning by tinkering with the Modified National Institute of Standards and Technology (MNIST) database of handwritten digits, which is commonly used to train image-processing systems. One of the seminal publications in probabilistic programming came in 2015 from Brenden Lake, Ruslan Salakhutdinov and Joshua Tenenbaum, who described a computational model that can recognize, parse and re-create these handwritten characters.[4] Their program learns quickly from limited samples and has an uncanny ability to use what it has already learned to create further new semantics, or meanings, much as a person would.

Their framework draws from three key areas in machine learning: compositionality, causality and "learning to learn." Rich concepts are constructed compositionally from primitive elements (the most basic elements available in the programming language), which serve as building blocks. Probabilistic semantics that are robust to noise — that is, are stable despite the addition of random elements — help this framework capture the causality inherent in real-world processes. The model learns to learn by using previous experience with related concepts to facilitate the mastery of new concepts, thereby constructing programs that best explain observations under a Bayesian criterion.

The technology seems to be best suited to making judgments in the presence of uncertainty, just as traditional computing technology works best as large-scale recordkeeping.

These priors embody a learned inductive bias — a set of assumptions that the learner uses to predict outputs, given inputs that it has not yet encountered — that captures key regularities and variations across concepts, and instances of these concepts in a given data domain.[5] New programs can be constructed reusing pieces of existing ones, exploiting the causal and compositional properties of real-world generative processes operating on multiple scales. New program types can be broadly generated by choosing primitive actions from a library and combining these subparts into larger components and relationships to define simple programs. Running these programs and rendering them as raw data can generate new program types.

THE VIRTUES OF BAYESIAN PROGRAM SYNTHESIS

One of the strengths of Bayesian probabilistic programs is that they are capable of writing new Bayesian programs themselves, with no need for human programmers. This so-called Bayesian program synthesis (BPS) framework separates modeling and inference, allowing the automatic synthesis of new programs via inference; the programs then feed into a modular framework and are integrated into larger models.

Because Bayesian methods easily incorporate prior knowledge, effective inference can often be performed with a lot less data. This increases the efficiency of implementing new models and understanding data. Just as high-level programming languages transformed developer productivity by eliminating the need to deal with technical details of processors and memory architecture, probabilistic languages promise to free developers from the complexities of high-performance probabilistic inference. In short, BPS allows more efficient search to find the optimum possibility. The constraints are probabilistic, and the output is a probability distribution that can then be further refined.

Another way to look at BPS is that probabilistic programs turn a simulation problem into an inference program. In other words, simulation requires certain rules to arrive at an estimated probability, whereas inference uses that probability data to arrive at a set of rules. The cycling between these two classes is a learning process (see Figure 1). The technology seems to be best suited to making judgments in the presence of uncertainty, just as traditional computing technology works best as large-scale recordkeeping.

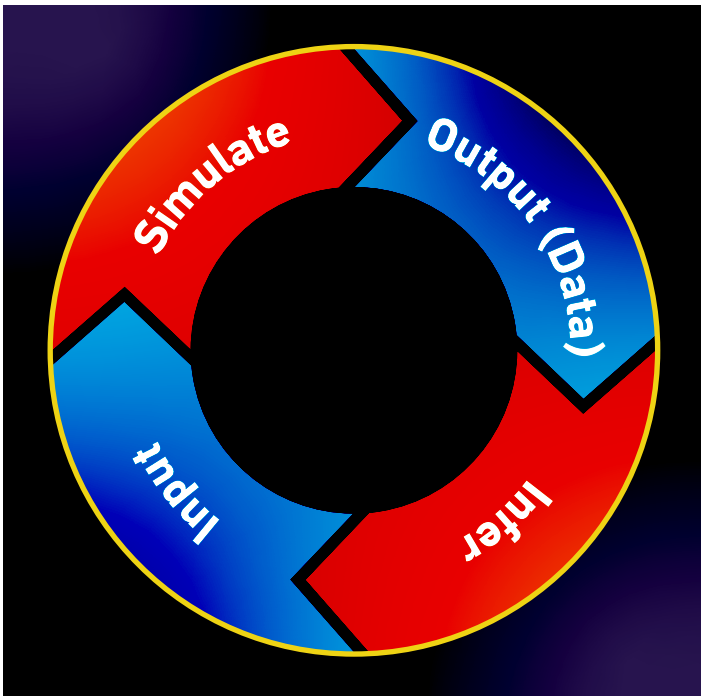

Figure 1: The Bayesian program synthesis uses rules to produce probabilistic data, which is then processed by inference programs to generate a new set of rules.

Unlike current computers built for logical deduction and precise arithmetic, the machines and programs of probabilistic computing are designed to handle ambiguity and learn from experience.

BPS AND DEEP LEARNING

If programs can take suggestions instead of blindly relying on data, machines may be able to act more like humans. For all its successes, deep learning still requires vast computational resources and ends up with a black-box function that is not transparent to humans. The BPS approach of automatically combining and modifying simple program pieces, incorporating human input and domain knowledge, can carry out much more complex tasks. Embedding the classic scientific method of hypothesize-test-iterate into a Bayesian framework and narrowing the possibilities based on new evidence allow this iterative process to be self-correcting and can yield even better results than classical deep learning. BPS points to the benefits of modeling underlying causal processes in learning concepts, a strategy different from some deep learning approaches. The other key ingredients — compositionality and learning to learn — also enhance the humanlike generative proficiency of Bayesian programming.

The difference between BPS and deep learning lies in the potential to analyze and interpret the models learned. Modern neural networks such as TensorFlow contain billions of neurons whose relationships to one another are constantly refined as the system strives for ever greater accuracy. During training, these values are adjusted up and down to give better predictions, but in the end a string of a billion numbers is impossible to read and messy, real-world training data can teach machines unintended lessons, as Google was surprised to learn when its image-search program concluded that dumbbells come with human arms attached.

Bayesian program synthesis works differently. Rather than adjusting decimal numbers, it modifies code in its simulation to try to match its predictions with those of a human. If we go into the system to find out what the program has learned, we can see exactly what the system is thinking. This open dialogue between man and machine lets developers quickly update the code with new rules of thumb. There is no way to tell TensorFlow to obey a certain rule, because you cannot go in and adjust the billions of synapse numbers. The BPS system, however, allows a program or human to easily add assumptions to a model, then test them on the data.

R&D IN PROBABILISTIC COMPUTING

Probabilistic computing is seeing renewed research at tech companies like Intel Corp., where scientists are working to find ways to build semiconductor chips that can handle errors and, to a certain extent, imprecision.[6]

Intel is betting big on probabilistic computing as a major component of AI that will allow systems to use natural data, with all its inherent uncertainties, and researchers to build computers capable of understanding, predicting and making decisions. Today a key barrier to AI is that natural data fed to a computer is largely unstructured and noisy. Probabilistic computing is efficient at dealing with probabilities on a scale that can transform current systems and applications from advanced computational aids into intelligent partners for understanding and decision-making.[7] Industry research has allocated substantial resources to benchmark applications for probabilistic frameworks. Academic research into new hardware is also gaining traction.

Probabilistic computing is paving the way for new architectures that can help optimize speed on basic algorithms. BPS represents a huge opportunity to explore a richer space of algorithms. Probabilistic computation also allows an increase in speed that is orders of magnitude faster and offers a substantial reduction

Intel is betting big on probabilistic computing as a major component of AI that will allow systems to use natural data, with all its inherent uncertainties.

Probabilistic computing is paving the way for new architectures that can help optimize speed on basic algorithms.

in power consumption. Currently, the AI hardware market has seen companies like Apple and Google promoting their specialized hardware to accelerate tasks like computer vision and image recognition, but AI and data analytics are no longer restricted to the big technology firms.

Many start-ups are investing in hardware and providing stiff competition to graphics processing units (GPUs), the current backbone of intensive computational applications for AI-related technologies. Practically every large tech company has invested in GPUs to fast-track its work around deep learning applications and to train complex models. Although the speed competition between GPUs and CPUs (central processing units) has been widely publicized, GPUs are ten times more efficient than CPUs in terms of power consumption. As barriers to CPU scaling rise with each successive shrinkage in nodes, the number of computer scientists seeking alternate methods of driving higher performance and/or saving power has grown.

NEW AI CAPABILITIES

Probabilistic computing will allow future systems to deal with the uncertainties inherent in natural data, enabling the development of computers capable of understanding, predicting and making decisions. For years, AI researchers have drawn inspiration from what is known about the human brain, and they have enjoyed considerable success as a result. Now AI is returning the favor. Current work may shed light on the neural representations of concepts and the development of more neurally grounded learning models.

This will open a path to a new generation of computing systems that will integrate probability and randomness into the basic building blocks of software and hardware.

Probabilistic approaches will lead to surprising new AI capabilities. A computer can learn about a user's interests without requiring an impractical amount of data or hours of training. If machine learning can be done efficiently on a user's smartphone or laptop, personal data might not need to be shared with large companies. A robot or a self-driving car could learn about a new obstacle without requiring hundreds or thousands of examples. In the domain of speech, programs for spoken words could be constructed by composing phonemes (subparts) systematically to form syllables (parts), which in turn could produce entire words. The one-shot learning capabilities studied by Lake, Salakhutdinov and Tenenbaum are a challenge for neural models and one they might rise to by incorporating the principles of compositionality, causality and learning to learn that BPL represents. ■

Michael Kozlov is a Senior Executive Research Director at WorldQuant, LLC, and has a Ph.D. in theoretical physics from Tel Aviv University.

Ashish Kulkarni is a Vice President, Research, at WorldQuant, LLC, and has a master's in information systems from MIT and an MS in molecular dynamics form Penn State University.

ENDNOTES

1. Fei Xu and Joshua B. Tenenbaum. "Word Learning as Bayesian Inference." *Psychological Review* 114 no. 2 (2007): 245–272.
2. Stuart Geman, Ellie Bienenstock and René Doursat. "Neural Networks and the Bias/Variance Dilemma." *Neural Computation* 4, no. 1 (1992): 1–58.
3. Alan Jern and Charles Kemp. "A Probabilistic Account of Exemplar and Category Generation." *Cognitive Psychology* 66, no. 1 (2013): 85–125.
4. Brenden M. Lake, Ruslan Salakhutdinov and Joshua B. Tenenbaum. "Human-Level Concept Learning Through Probabilistic Program Induction." *Science* 350, no. 6266 (2015): 1332–1338.
5. Vladimir Naumovich Vapnik. "An Overview of Statistical Learning Theory." *IEEE Transactions on Neural Networks* 10, no. 5 (1999): 988–999.
6. Michael Mayberry. "Probabilistic Computing Takes Artificial Intelligence to the Next Step." Intel Newsroom, May 10, 2018.
7. Samuel K. Moore. "Intel Starts R&D Effort in Probabilistic Computing for AI." *IEEE Spectrum*, May 10, 2018