Deep Learning's DIMINISHING RETURNS

The cost of improvement is becoming unsustainable

BY NEIL C. THOMPSON, KRISTJAN GREENEWALD, KEEHEON LEE & GABRIEL F. MANSO

DEEP LEARNING IS NOW being used to translate between languages, predict how proteins fold, analyze medical scans, and play games as complex as Go, to name just a few applications of a technique that is now becoming pervasive. Success in those and other realms has brought this machine-learning technique from obscurity in the early 2000s to dominance today. • Although deep learning's rise to fame is relatively recent, its origins are not. In 1958, back when mainframe computers filled rooms and ran on vacuum tubes, knowledge of the interconnections between neurons in the brain inspired Frank Rosenblatt at Cornell to design the first artificial neural network, which he presciently described as a "pattern-recognizing device." But Rosenblatt's ambitions outpaced the capabilities of his era—and he knew it. Even his inaugural paper was forced to acknowledge the voracious appetite of neural networks for computational power, bemoaning that "as the number of connections in the network increases...the burden on a conventional digital computer soon becomes excessive." • Fortunately for such artificial neural networks—later rechristened "deep learning" when

Illustration by Eddie Guy

they included extra layers of neurons—decades of Moore's Law and other improvements in computer hardware yielded a roughly 10-million-fold increase in the number of computations that a computer could do in a second. So when researchers returned to deep learning in the late 2000s, they wielded tools equal to the challenge.

These more-powerful computers made it possible to construct networks with vastly more connections and neurons and hence greater ability to model complex phenomena. Researchers used that ability to break record after record as they applied deep learning to new tasks.

While deep learning's rise may have been meteoric, its future may be bumpy. Like Rosenblatt before them, today's deep-learning researchers are nearing the frontier of what their tools can achieve. To understand why this will reshape machine learning, you must first understand why deep learning has been so successful and what it costs to keep it that way.

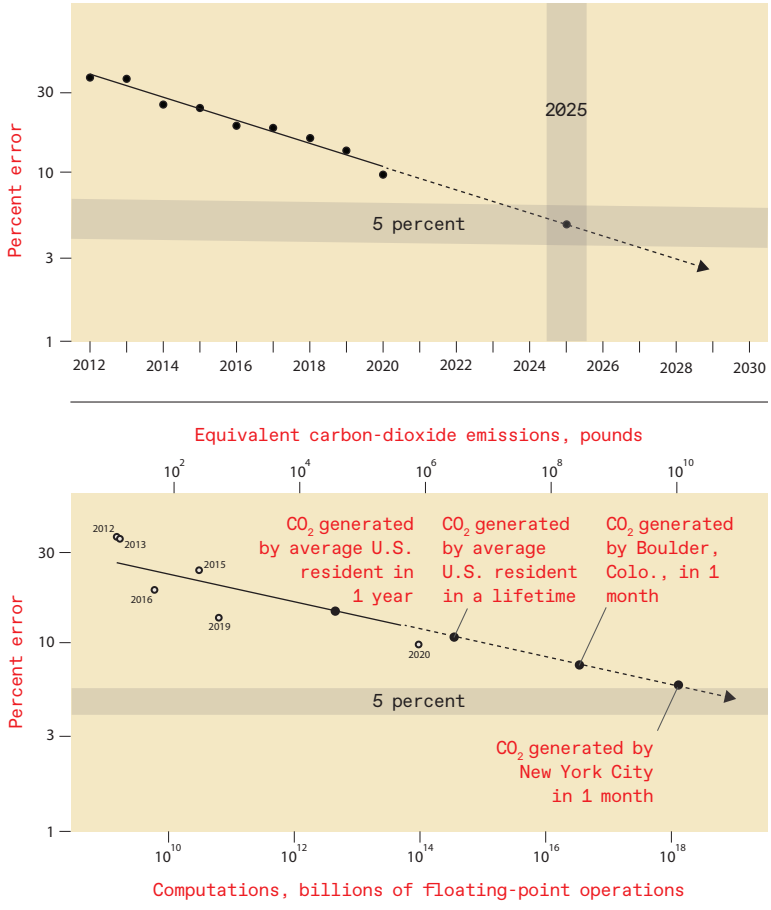
DEEP LEARNING is a modern incarnation of the long-running trend in artificial intelligence that has been moving from streamlined systems based on expert knowledge toward flexible statistical models. Early AI systems were rule based, applying logic and expert knowledge to derive results. Later systems incorporated learning to set their adjustable parameters, but these were usually few in number.

Today's neural networks also learn parameter values, but those parameters are part of such flexible computer models that—if they are big enough—they become universal function approximators, meaning they can fit any type of data. This unlimited flexibility is the reason why deep learning can be applied to so many different domains.

The flexibility of neural networks comes from taking the many inputs to the model and having the network combine them in myriad ways. This means the outputs won't be the result of applying simple formulas but instead immensely complicated ones.

For example, when the cutting-edge image-recognition system Noisy Student converts the pixel values of an image into probabilities for what the object in that image is, it does so using a network with 480 million parameters. The training to ascertain the values of such a large number of parameters is even more remarkable because it was done with only 1.2 million labeled images—which may understandably confuse those of us who remember from high school algebra that we are supposed to have more equations than unknowns. Breaking that rule turns out to be the key.

Deep-learning models are overparameterized, which is to say they have more parameters than there are data points available for training. Classically, this would lead to overfitting, where the model not only learns general trends but also the random vagaries of the data it was trained on. Deep learning avoids this trap by initializing the parameters randomly and then iteratively adjusting sets of them to better fit the



Extrapolating the gains of recent years might suggest that by 2025 the error level in the best deep-learning systems designed for recognizing objects in the ImageNet data set should be reduced to just 5 percent [top]. But the computing resources and energy required to train such a future system would be enormous, leading to the emission of as much carbon dioxide as New York City generates in one month [bottom].

SOURCE: N.C. THOMPSON, K. GREENEWALD, K. LEE, G.F. MANSO

data using a method called stochastic gradient descent. Surprisingly, this procedure has been proven to ensure that the learned model generalizes well.

The success of flexible deep-learning models can be seen in machine translation. For decades, software has been used to translate text from one language to another. Early approaches to this problem used rules designed by grammar experts. But as more textual data became available in specific languages, statistical approaches—ones that go by such esoteric names as maximum entropy, hidden Markov models, and conditional random fields—could be applied.

Initially, the approaches that worked best for each language differed based on data availability and grammatical properties. For example, rule-based approaches to translating languages such as Urdu, Arabic, and Malay outperformed statistical ones—at first. Today, all these approaches have been outpaced by deep learning, which has proven itself superior almost everywhere it's applied.

So the good news is that deep learning provides enormous flexibility. The bad news is that this flexibility comes at an enormous computational cost. This unfortunate reality has two parts.

The first part is true of all statistical models: To improve performance by a factor of k, at least k^2 more data points must be used to train the model. The second part of the computational cost comes explicitly from overparameterization. Once accounted for, this yields a total computational cost for improvement of *at least* k^4. That little 4 in the exponent is very expensive: A 10-fold improvement, for example, would require at least a 10,000-fold increase in computation.

To make the flexibility-computation trade-off more vivid, consider a scenario where you are trying to predict whether a patient's X-ray reveals cancer. Suppose further that the true answer can be found if you measure 100 details in the X-ray (often called variables or features). The challenge is that we don't know ahead of time which variables are important, and there could be a very large pool of candidate variables to consider.

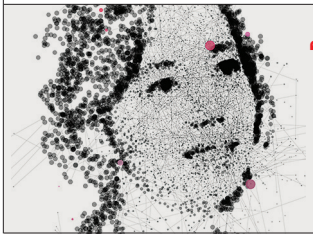
The expert-system approach to this problem would be to have people who are knowledgeable in radiology and oncology specify the variables they think are important, allowing the system to examine only those. The flexible-system approach is to test as many of the variables as possible and let the system figure out on its own which are important, requiring more data and incurring much higher computational costs in the process.

Models for which experts have established the relevant variables are able to learn quickly what values work best for those variables, doing so with limited amounts of computation—which is why they were so popular early on. But their ability to learn stalls if an expert hasn't correctly specified all the variables that should be included in the model. In contrast, flexible models like deep learning are less efficient, taking vastly more computation to match the performance of expert models. But, with enough computation (and data), flexible models can outperform ones for which experts have attempted to specify the relevant variables.

CLEARLY, YOU CAN GET improved performance from deep learning if you use more computing power to build bigger models and train them with more data. But how expensive will this computational burden become? Will costs become sufficiently high that they hinder progress?

To answer these questions in a concrete way, we recently gathered data from more than 1,000 research papers on deep learning, spanning the areas of image classification, object detection, question answering, named-entity recognition, and machine translation. Here, we will only discuss image classification in detail, but the lessons apply broadly.

Over the years, reducing image-classification errors has come with an enormous expansion in computational burden. For example, in 2012 AlexNet, the model that first showed the power of training deep-learning systems on graphics processing units (GPUs), was trained for five to six days using two GPUs. By 2018, another model, NASNet-A,



PORTRAIT BY SERGIO ALBIAC

"AI is fundamentally an applied technology that's going to serve our society. Humanistic AI not only raises the awareness of the importance of the technology, it's a really important way to attract diverse students, technologists, and innovators to participate."

FEI-FEI LI, codirector of the Stanford Institute for Human-Centered AI

had cut the error rate of AlexNet in half, but it used more than 1,000 times as much computing to achieve this.

Our analysis of this phenomenon also allowed us to compare what actually happened with theoretical expectations. Theory tells us that computing needs to scale with at least the fourth power of the improvement in performance. In practice, the actual requirements have scaled with at least the *ninth* power.

This ninth power means that to halve the error rate, you can expect to need more than 500 times the computational resources. That's a devastatingly high price. There may be a silver lining here, however. The gap between what's happened in practice and what theory predicts might mean that there are still undiscovered algorithmic improvements that could greatly improve the efficiency of deep learning.

As we noted, Moore's Law and other hardware advances have provided massive increases in chip performance. Does this mean that the escalation in computing requirements doesn't matter? Unfortunately, no. Of the 1,000-fold difference in the computing used by AlexNet and NASNet-A, only a sixfold improvement came from better hardware; the rest came from using more processors or running them longer, incurring higher costs.

Having estimated the computational cost-performance curve for image recognition, we can use it to estimate how much computation would be needed to reach even more impressive performance benchmarks in the future. For example, achieving a 5 percent error rate would require 10^{19} billion floating-point operations.

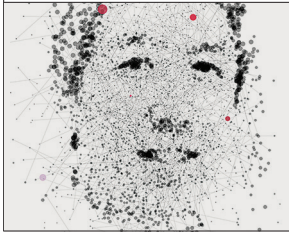
Important work by scholars at the University of Massachusetts Amherst allows us to understand the economic cost and carbon emissions implied by this computational burden. The answers are grim: Training such a model would cost US $100 billion and would produce as much carbon emissions as New York City does in a month. And if we estimate the computational burden of a 1 percent error rate, the results are considerably worse.

Is extrapolating out so many orders of magnitude a reasonable thing to do? Yes and no. Certainly, it is important to understand that the predictions aren't precise, although with such eye-watering results, they don't need to be to convey the overall message of unsustainability. Extrapolating this way *would* be unreasonable if we assumed that researchers would follow this trajectory all the way to such an extreme outcome. We don't. Faced with skyrocketing costs, researchers will either have to come up with more efficient ways to solve these problems, or they will abandon working on these problems and progress will languish.

On the other hand, extrapolating our results is not only reasonable but also important, because it conveys the magnitude of the challenge ahead. The leading edge of this problem is already becoming apparent. When Google subsidiary DeepMind trained its system to play Go, it was estimated to have cost $35 million. When DeepMind's researchers designed a system to play the *StarCraft II* video game, they purposefully didn't try multiple ways of architecting an important component, because the training cost would have been too high.

At OpenAI, an important machine-learning think tank, researchers recently designed and trained a much-lauded deep-learning language system called GPT-3 at the cost of more than $4 million. Even though they made a mistake when they implemented the system, they didn't fix it, explaining simply in a supplement to their scholarly publication that "due to the cost of training, it wasn't feasible to retrain the model."

Even businesses outside the tech industry are now starting to shy away from the computational expense of deep learning. A large European supermarket chain recently abandoned a deep-learning-based system that markedly improved its ability to predict which products would be purchased. The company executives dropped that attempt because they judged that the cost of training and running the system would be too high.



"Those of us in machine learning are really good at doing well on a test set, but unfortunately deploying a system takes more than doing well on a test set. All of AI...has a proof-of-concept-to-production gap." **ANDREW NG**, CEO and cofounder of Landing AI

PORTRAIT BY SERGIO ALBIAC

FACED WITH RISING economic and environmental costs, the deep-learning community will need to find ways to increase performance without causing computing demands to go through the roof. If they don't, progress will stagnate. But don't despair yet: Plenty is being done to address this challenge.

One strategy is to use processors designed specifically to be efficient for deep-learning calculations. This approach was widely used over the last decade, as CPUs gave way to GPUs and, in some cases, field-programmable gate arrays and application-specific ICs (including Google's Tensor Processing Unit). Fundamentally, all of these approaches sacrifice the generality of the computing platform for the efficiency of increased specialization. But such specialization faces diminishing returns. So longer-term gains will require adopting wholly different hardware frameworks—perhaps hardware that is based on analog, neuromorphic, optical, or quantum systems. Thus far, however, these wholly different hardware frameworks have yet to have much impact.

Another approach to reducing the computational burden focuses on generating neural networks that, when implemented, are smaller. This tactic lowers the cost each time you use them, but it often increases the training cost (what we've described so far in this article). Which of these costs matters most depends on the situation. For a widely used model, running costs are the biggest component of the total sum invested. For other models—for example, those that frequently need to be retrained—training costs may dominate. In either case, the total cost must be larger than just the training on its own. So if the training costs are too high, as we've shown, then the total costs will be, too.

And that's the challenge with the various tactics that have been used to make implementation smaller: They don't reduce training costs enough. For example, one allows for training a large network but penalizes complexity during training. Another involves training a large network and then "prunes" away unimportant connections. Yet another finds as efficient an architecture as possible by optimizing across many models—something called neural-architecture search. While each of these techniques can offer significant benefits for implementation, the effects on training are muted—certainly not enough to address the concerns we see in our data. And in many cases they make the training costs higher.

One up-and-coming technique that could reduce training costs goes by the name meta-learning. The idea is that the system learns on a variety of data and then can be applied in many areas. For example, rather than building separate systems to recognize dogs in images, cats in images, and cars in images, a single system could be trained on all of them and used multiple times.

We must either adapt how we do deep learning or face a future of much slower progress.

Unfortunately, recent work by Andrei Barbu of MIT has revealed how hard meta-learning can be. He and his coauthors showed that even small differences between the original data and where you want to use it can severely degrade performance. They demonstrated that current image-recognition systems depend heavily on things like whether the object is photographed at a particular angle or in a particular pose. So even the simple task of recognizing the same objects in different poses causes the accuracy of the system to be nearly halved.

Benjamin Recht of the University of California, Berkeley, and others made this point even more starkly, showing that even with novel data sets purposely constructed to mimic the original training data, performance drops by more than 10 percent. If even small changes in data cause large performance drops, the data needed for a comprehensive meta-learning system might be enormous. So the great promise of meta-learning remains far from being realized.

Another possible strategy to evade the computational limits of deep learning would be to move to other, perhaps as-yet-undiscovered or underappreciated types of machine learning. As we described, machine-learning systems constructed around the insight of experts can be much more computationally efficient, but their performance can't reach the same heights as deep-learning systems if those experts cannot distinguish all the contributing factors. Neuro-symbolic methods and other techniques are being developed to combine the power of expert knowledge and reasoning with the flexibility often found in neural networks.

Like the situation that Rosenblatt faced at the dawn of neural networks, deep learning is today becoming constrained by the available computational tools. Faced with computational scaling that would be economically and environmentally ruinous, we must either adapt how we do deep learning or face a future of much slower progress. Clearly, adaptation is preferable. A clever breakthrough might find a way to make deep learning more efficient or computer hardware more powerful, which would allow us to continue to use these extraordinarily flexible models. If not, the pendulum will likely swing back toward relying more on experts to identify what needs to be learned. ■