Engineering 28 (2023) 1–3

Contents lists available at ScienceDirect

Engineering

journal homepage: www.elsevier.com/locate/eng

News & Highlights

After Artificial Intelligence Breaks Longstanding Matrix Multiplication Records, Humans Quickly Do Better

Dana Mackenzie

Senior Technology Writer

In October 2022, machine learning experts at DeepMind (London, UK), a subsidiary of Google (Mountain View, CA, USA), reported a "breakthrough" on an extremely common mathematical algorithm called matrix multiplication [1]. In previous years, DeepMind has made headlines with its successes in using deep learning to master various games, such as Go [2], chess, and even the strategic board game Diplomacy, and, in a more recent and clearly practical application with its AlphaFold program [3], to spawn a revolution in structural biology [4,5]. Now, the company claimed, its AlphaTensor program had used reinforcement learning to turn matrix multiplication into a "game" and discovered methods to do it in fewer steps than any human had found, improving upon a record in one case that had stood for more than 50 years.

Among experts on fast matrix multiplication, however, the reaction has been mixed. "While the AlphaTensor work has not advanced our scientific knowledge, it has generated interest in this fundamental problem," said J. M. Landsberg, professor of mathematics at Texas A&M University in College Station, TX, USA. Other experts are not so critical but said it is still far from certain that the new algorithms discovered by AlphaTensor will make any practical difference. It is impressive as a demonstration of the capabilities of artificial intelligence (AI), but not particularly impressive as an advance in mathematics. "That is the point—they are trying to show just how powerful AI is," said Landsberg. DeepMind declined multiple requests to discuss the work.

"Matrix multiplication" refers to a way of multiplying two arrays of numbers (matrices) to get a third matrix. It is a ubiquitous operation in science and engineering, used to solve systems of linear equations and often to approximate solutions to nonlinear equations, such as those that occur in computer simulations of the climate, galaxies, or internal combustion engines. Indeed, it is like a piston of computer science: a tool used so often it is sometimes taken for granted. Most mathematicians and computer scientists learn a standard procedure for multiplying matrices when they are university students. The procedure is a couple hundred years old, but it was only in 1969 that Volker Strassen, a German mathematician, discovered a better way to do it [6].

Better, in this case, means a way that involves less multiplication. For example, when multiplying a 2-by-2 matrix by a 2-by-2 matrix, the standard method requires you to multiply eight different numbers together. There are also sums involved, but in a computer, multiplication is the more time-consuming operation. Strassen, very surprisingly, discovered a way to multiply 2-by-2 matrices that requires only seven multiplications rather than eight (Fig. 1). Even better, his method scales up very nicely. If you want to multiply two 4-by-4 matrices, all you must do is divide them into 2-by-2 blocks. Applying Strassen's formula to a 4-by-4 matrix reduces the computation to seven products of 2-by-2 matrices. Applying it again to compute each of these products makes it 49 multiplications in all—a significant improvement over the 64 multiplications required by the textbook method for 4-by-4 matrices. Using this iterative procedure, the number of operations to multiply two *N*-by-*N* matrices can be reduced to *N* to the 2.8074 power, a significant speedup compared to the textbook algorithm (which requires *N* to the third power steps).

The exponent is important, said Landsberg. "It is a fundamental constant of nature concerning how fast we can multiply matrices. It is either equal to 2 or it is not 2. If it is 2, that basically means that as matrices get large, they are almost as easy to multiply as to add." So far, mathematicians have whittled the number of steps down to *N* to the 2.37188 power [7], using complicated algebraic methods that involve a great deal of computer calculation but no automated discovery. That was a tiny improvement over the

Practical fast matrix multiplication

$$\begin{aligned}
M_1 &= (A_{11}+A_{22})\cdot(B_{11}+B_{22})\\
M_2 &= (A_{21}+A_{22})\cdot B_{11}\\
M_3 &= A_{11}\cdot(B_{12}-B_{22})\\
M_4 &= A_{22}\cdot(B_{21}-B_{11})\\
M_5 &= (A_{11}+A_{22})\cdot B_{22}\\
M_6 &= (A_{21}-A_{11})\cdot(B_{11}+B_{12})\\
M_7 &= (A_{12}-A_{22})\cdot(B_{21}+B_{22})
\end{aligned}$$

$$\begin{aligned}
C_{11} &= M_1+M_4-M_5+M_7\\
C_{12} &= M_3+M_5\\
C_{21} &= M_2+M_4\\
C_{22} &= M_1-M_2+M_3+M_6
\end{aligned}$$

$$U=\begin{pmatrix}1&0&1&0&1&-1&0\\0&0&0&0&1&0&1\\0&1&0&0&0&1&0\\1&1&0&1&0&0&-1\end{pmatrix}$$

$$V=\begin{pmatrix}1&1&0&-1&0&1&0\\0&0&1&0&0&1&0\\0&0&0&1&0&0&1\\1&0&-1&0&1&0&1\end{pmatrix}$$

$$W=\begin{pmatrix}1&0&0&1&-1&0&1\\0&1&0&1&0&0&0\\0&0&1&0&1&0&0\\1&-1&1&0&0&1&0\end{pmatrix}$$

Fig. 1. Left: Strassen's algorithm for finding the product *C* of two 2-by-2 matrices (*A* and *B*). Right: A compact representation of the algorithm as a 3-dimensional tensor, a stack of three matrices, *U*, *V*, and *W*, which serves as a bookkeeping device to show which products to write in each of the four slots in the 2-by-2 matrix of product *C*. The key point is these matrices have only seven columns, with only seven products to add up rather than the eight used in the textbook formula for calculating *C*. Credit: Grey Ballard, with permission.

https://doi.org/10.1016/j.eng.2023.07.002

previous best estimate, 2.37286—which gives you an idea of how slow progress has been and how far we are from reducing the exponent all the way to 2. Progress in the other direction—proving that the constant is greater than 2 (if it is)—has been completely nonexistent. One book calls this type of problem "complexity theory's Waterloo" [8].

Hence the extraordinary interest in DeepMind's AlphaTensor work. First, it gives a new paradigm for finding multiplication algorithms: Do not bother teaching the computer math, just teach it how to play games. Math is just another game. And second, once a new algorithm is found, anybody can use it, even if they do not have DeepMind's giant computer resources

But is it a genuine breakthrough? The answer is complicated. First, even Strassen's algorithm is not as widely used as one might expect. One reason is that it is not error-proof, especially if some entries in the product matrix are zero or near zero. (The errors can even be larger than the entry itself.) "This scares people off," said Grey Ballard, associate professor in computer science at Wake Forest University in Winston-Salem, NC, USA. "I think they worry more than they should, however, because they are doing other computations that have less accuracy. Maybe the most compelling example is deep learning: There are errors in the data, errors in the iterative algorithm, and errors within the iteration. The requirements of matrix multiplication accuracy in this application are very low."

Another obstacle is sociological. Matrix multiplication is used as a benchmark, one of a series of measurements that are used to evaluate how fast a computer is. These benchmarks carry enormous prestige for makers of supercomputers [9]. "The benchmark outlaws the use of fast matrix multiplication," Ballard said. "It is a reasonable restriction because they want to benchmark the machine, not the algorithm." But the hidden cost is that programmers devote all their efforts to fine-tuning the standard algorithm to run faster on their computer, rather than developing new fast algorithms.

Despite these obstacles, a major improvement to fast matrix multiplication would be expected to quickly make inroads against the standard algorithm. So far, AlphaTensor has modestly improved the procedure for multiplying two matrices in five cases, two of which come with huge caveats. For 4-by-4 matrices, it can compute the product in 47 steps instead of 49. But the new formula only works to the extent of telling whether each of the entries in the product matrix is even or odd. If you want actual numerical values for your matrix product, rather than just "characteristic two" values (i.e., even, or odd), you still must use 49 multiplications and there is no improvement at all over the Strassen method.

The next case where AlphaTensor improved on the state of the art is the case of multiplying a 4-by-5 matrix by a 5-by-4 matrix. Here the program reduced the previous record from 64 multiplications to 63. However, this achievement soon became moot. Within a week, Professor Manuel Kauers and PhD student Jakob Moosbauer at Johannes Kepler University in Linz, Austria, achieved a new record of 62 multiplications [10]. The work was performed on a small computer with a human-devised search method they called "flip graphs." Though computer based, their search method does not require anything resembling machine learning or AI. It does require a good starting point, because it is based on a random walk through the space of possible algorithms, and it helps to start with an algorithm that is already quite efficient.

Finally, in the case of 5-by-5 times 5-by-5 matrices, AlphaTensor brought the record down from 98 to 96 multiplications, again only if one only cares about whether the matrix entries are even or odd. In this case, too, Kauers and Moosbauer surpassed the DeepMind work, with a method requiring only 95 multiplications. They commented that it took their small computer only a few seconds to run the search but acknowledged the advantage of using the DeepMind algorithm as a starting point.

In short, the DeepMind work has not exactly proved that machine learning is a better way to identify methods for more efficiently multiplying matrices. Two of their five improvements were superseded within days, and two of their five improvements were limited to the characteristic-two case (determining whether each entry in the matrix product is even or odd), which is, at best, a niche application. "I do not know anyone who has taken their algorithms and implemented them," Ballard said. And in the realm of abstract theory, their work has not improved our knowledge of the "fundamental constant of nature" limiting how fast two matrices can be multiplied. It remains stuck at 2.37188, where it was before the DeepMind research.

However, it would be equally misleading to dismiss the work completely. One positive outcome is that the AlphaTensor work re-opens an approach to discovering new algorithms called "combinatorial search," which had been considered too inefficient because it does not scale well. The time spent searching for a better algorithm increases exponentially with the size of the matrix. "People have tried this, including me," said Ballard. "I am impressed that they can solve problems of this size using a combinatorial technique."

Also, where there is one fast matrix multiplication algorithm, there are almost always many. This means that different algorithms could be used on different hardware, taking advantage of whatever hidden strengths or weaknesses they might have. The reinforcement learning technique might enable a machine to discover the matrix multiplication method that works best for it.

Finally, there is a possibility that sounds almost like science fiction: The neural network could, in principle, use the new algorithm to redesign itself. Neural networks, such as AlphaTensor, are organized into layers of "neurons," each one feeding forward (or backward) to the adjacent ones (Fig. 2). If one layer is fully connected to the next, every node to every other node, then the method by which the network propagates information from one layer to the next is a matrix product. So AlphaTensor's algorithms could, in theory, be used to program the next generation of machines to learn faster than it does. "Whether matrix multiplication is a bottleneck depends on what kind of neural net you have," said Ballard. "I am sure that AlphaTensor has matrix multiplications in there. So, they can definitely use it to make better neural networks."

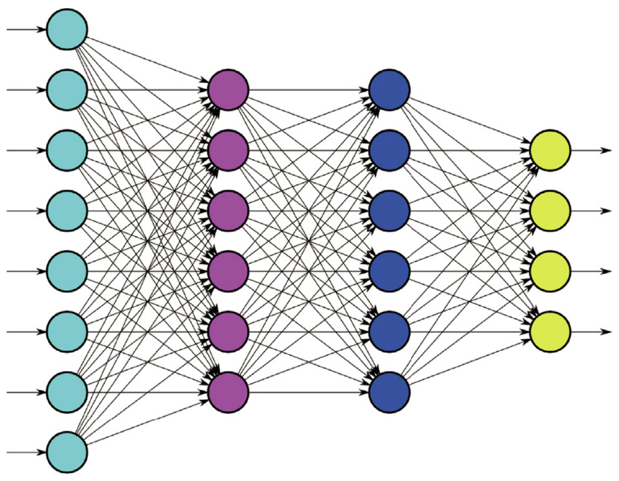
Fig. 2. A typical architecture for a neural network, with four layers of "neurons," an input layer (left), an output layer (right) and two hidden layers. In a fully connected neural network, all the nodes in one layer connect to all the nodes in the next. In this case, the procedure for computing one layer from the previous one begins by taking a matrix product. Credit: Heiko Loewe (public domain).

References

[1] Fawzi A, Balog M, Huang A, Hubert T, Romera-Paredes B, Barekatain M, et al. Discovering faster matrix multiplication algorithms with reinforcement learning. Nature 2022;610:47–52.
[2] Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, et al. Mastering the game of Go with deep neural networks and tree search. Nature 2016;529(7587):484–9.
[3] Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O, et al. Highly accurate protein structure prediction with AlphaFold. Nature 2021;596 (7873):583–9.
[4] O'Neill S. Artificial intelligence cracks a 50-year-old grand challenge in biology. Engineering 2021;7(6):706–8.
[5] O'Neill S. Machine learning turbocharges structural biology. Engineering 2022;12:9–11.
[6] Strassen V. Gaussian elimination is not optimal. Numer Math 1969;13(4):354–6.
[7] Duan R, Wu H, Zhou R. Faster matrix multiplication via asymmetric hashing. 2023. arXiv:2210.10173v4.
[8] Arora S, Barak B. Computational complexity: a modern approach. Cambridge: Cambridge University Press; 2009. p. 286.
[9] Palmer J. More super supercomputers. Engineering 2019;5(3):357–8.
[10] Kauers M, Moosbauer J. Flip graphs for matrix multiplication; 2022. arXiv:2212.01175v1.