IBM has made a new, highly efficient AI processor

reader comments

21 with

As the utility of AI systems has grown dramatically, so has their energy demand. Training new systems is extremely energy intensive, as it generally requires massive data sets and lots of processor time. Executing a trained system tends to be much less involved—smartphones can easily manage it in some cases. But, because you execute them so many times, that energy use also tends to add up.

Fortunately, there are lots of ideas on how to bring the latter energy use back down. IBM and Intel have experimented with processors designed to mimic the behavior of actual neurons. IBM has also tested executing neural network calculations in phase change memory to avoid making repeated trips to RAM.

Now, IBM is back with yet another approach, one that's a bit of "none of the above." The company's new NorthPole processor has taken some of the ideas behind all of these approaches and merged them with a very stripped-down approach to running calculations to create a highly power-efficient chip that can efficiently execute inference-based neural networks. For things like image classification or audio transcription, the chip can be up to 35 times more efficient than relying on a GPU.

A very unusual processor

It’s worth clarifying a few things early here. First, NorthPole does nothing to help the energy demand in training a neural network; it’s purely designed for execution. Second, it is not a general AI processor; it’s specifically designed for inference-focused neural networks. As noted above, inferences include things like figuring out the contents of an image or audio clip so they have a large range of uses, but this chip won’t do you any good if your needs include running a large language model.

Finally, while NorthPole takes some ideas from neuromorphic computing chips, including IBM’s earlier TrueNorth, this is not neuromorphic hardware, in that its processing units perform calculations rather than attempt to emulate the spiking communications that actual neurons use.

Advertisement

That’s what it’s not. What actually is NorthPole? Some of the ideas do carry forward from IBM’s earlier efforts. These include the recognition that a lot of the energy costs of AI come from the separation between memory and execution units. Since a key component of neural networks—the weight of connections between different layers of “neurons”—is held in memory, any execution on a traditional processor or GPU burns a lot of energy simply getting those weights from memory to where they can be used during execution.

So NorthPole, like TrueNorth before it, consists of a large array (16×16) of computational units, each of which includes both local memory and code execution capacity. So, all of the weights of various connections in the neural network can be stored exactly where they’re needed.

Another feature is extensive on-chip networking, with at least four distinct networks. Some of these carry information from completed calculations to the compute units where they’re

needed next. Others are used to reconfigure the entire array of compute units, providing the neural weights and code needed to execute one layer of the neural network while the calculations of the previous layer are still in progress. Finally, communications among neighboring compute units is optimized. This can be useful for things like finding the edge of an object in an image. If the image is entered so that neighboring pixels go to neighboring compute units, they can more easily cooperate to identify features that extend across neighboring pixels.

The computing resources are unusual as well. Each unit is optimized for performing lower-precision calculations, ranging from two- to eight-bit precision. While higher precision is often required for training, the values needed during execution generally don’t require that level of exactitude. To keep those execution units in use, they are incapable of performing conditional branches based on the value of variables—meaning your code cannot contain an “if” statement. This eliminates the need for the hardware needed for speculative branch execution, and it ensures that the wrong code will be executed whenever that speculation turns out to be wrong.

This simplicity in execution makes each compute unit capable of massively parallel execution. At two-bit precision, each unit can perform over 8,000 calculations in parallel.

https://arstechnica.com/?p=1977529