Accelerating Distributed Machine Learning on Standard Ethernet Infrastructure with SwitchML Technology

Intel is developing the industry’s first Ethernet-based, intelligent network switch that can help eliminate network communication bottlenecks, speeding up machine learning model training by up to 2.27x

Executive Summary

With billions of parameters and vast training datasets, today’s machine learning models can take days—or even weeks—to complete the training process. These long training times mean precious insights are missed or delayed. In many cases, the bottleneck is not caused by compute resources; instead, the suspect may be the significant network communication and infrastructure that are necessary to synchronize model updates across clusters with hundreds of nodes.

Intel, working with other ecosystem players such as Microsoft, the King Abdullah University of Science and Technology (KAUST), and the University of Washington, is developing an intelligent network technology to accelerate machine learning training, called SwitchML. Unlike other networking solutions that require specialized and proprietary protocols such as InfiniBand, SwitchML can be used on industry-standard Ethernet infrastructure, which offers several advantages:

- Interoperability with existing network components
- Freedom of choice for network component suppliers
- Ease of adoption

SwitchML works with popular artificial intelligence (AI) frameworks like TensorFlow and PyTorch, with more ecosystem integrations to come.
Business Challenge: Inefficient Network Communication Shackles ML Training

For many years, the complexity of machine learning models followed Moore’s Law, doubling about every two years (see the top portion of Figure 1). But more recently, as machine learning (ML) and deep learning (DL) mature, model complexity has skyrocketed. In fact, for some types of models, such as image recognition and natural language processing, the number of parameters in a model increases 10x every year (see the bottom portion of Figure 1). Some models, such as Google Brain’s Switch Transformer, have more than a trillion parameters. As models become more complex, they require larger datasets for training to increase the model’s accuracy. The larger the training dataset, the more accurate a model becomes. But such large training datasets require distributing the ML training over a cluster made up of hundreds of worker nodes. Even with clusters of this size, training can take weeks or months to complete.

In a distributed, data-parallel ML training cluster, each worker node learns a certain amount based on the training data, and must share what it learns with the other nodes (called the synchronization phase of distributed ML, illustrated in Figure 2). Typically, several hundred nodes must communicate their own learning to all the others; this synchronization phase occurs in each iteration, for many iterations.

Today’s distributed data-parallel ML clusters typically use one of two approaches to perform the synchronization phase. One approach is to use parameter servers; the other approach is to use collective operations, like Allreduce. In both cases, additional network resources and computation cycles are spent to perform this operation, leading to either an increase of the cost of the infrastructure or increased communication time. The synchronization phase requires transmitting hundreds of MBs or even GBs for every single node and for each iteration. The duration of a single iteration is typically hundreds of milliseconds, and the communication phase can take a substantial portion of this time.

It’s easy to assume that the bottleneck in such large, distributed clusters is caused by available compute resources. However, that is not necessarily true. Most ML training clusters are equipped with multiple high-
performance accelerators, such as graphics processing units (GPUs), field-programmable gate arrays (FPGAs), or dedicated application-specific integrated circuits (ASICs). On average, compute power increases according to Moore’s Law (at the rate of about 2x in two years). For many models, the real latency culprit is the network capacity and speed, which have not advanced nearly as fast as compute power. The result of the network lagging behind the compute power is that during the synchronization phase, much time is wasted in network communication, leaving the powerful—and expensive—GPUs, FPGAs, and ASICs sitting idle. To illustrate the problem, consider the results shown in Figure 3. Some models spend up to 80 percent of their runtime on communication, even with a high-speed 100 GbE network fabric, and a significant portion of that time does not overlap with computation. The results shown in Figure 3 underscore an important point: ML scales poorly due to communication overhead. These results also raise an interesting question: Can the network itself become the ML accelerator?

The SwitchML prototype can aggregate model updates directly inside the Ethernet network, making the network a new ML accelerator (see Figure 4). It is important to note that this solution can be incrementally deployed in existing data center networks, including support for heterogeneous networks.

Leading the Way: The First Ethernet-Based Programmable Switching Data Plane for In-Network Aggregation

Intel, working with Microsoft Research, the King Abdullah University of Science and Technology (KAUST), and the University of Washington, is designing a communication primitive that uses a P4-programmable switching data plane to execute a key step of the training process. This approach, called SwitchML technology, uses in-network aggregation to reduce the volume of data being transferred during network synchronization phases of training. This in turn reduces the overall latency in many-to-many communications in large networks. Tests show it can speed up training by up to 2.27x (on a 100 Gbps network). The SwitchML prototype can aggregate model updates directly inside the Ethernet network, making the network a new ML accelerator (see Figure 4). It is important to note that this solution can be incrementally deployed in existing data center networks, including support for heterogeneous networks.

SwitchML works with popular AI frameworks like TensorFlow and PyTorch; can be deployed seamlessly as an NVIDIA Collective Communications Library (NCCL) plugin; and provides easy-to-use APIs, allowing any software project to use it directly. And, because SwitchML is interoperable with common data center Ethernet network components, easy adoption is promising. Network engineers can choose from a wide variety of switch vendors to build their network, and can incrementally deploy it in existing networks, without having to change the entire infrastructure. The current prototype is designed to perform the common Allreduce collective. However, the programmability of the solution means that it could easily be adapted to support other collectives, such as Broadcast, Barrier, Allgather, Reduce, and ReduceScatter.
A Closer Look at the Innovative SwitchML Prototype

To implement our vision of an in-network Allreduce collective, we had to surmount a few challenges:

- **Limited storage.** The only available storage for programmable switch ASICs is the costly and scarce on-chip memory.

- **Limited computation.** Because programmable switch ASICs must process network traffic at a very high speed, they have only a few nanoseconds to process a single packet. Therefore, they cannot perform a large amount of computation.

- **Lack of floating-point units (FPUs).** Typically, programmable switch ASICs do not provide FPUs, because they are not usually needed by common network protocols. But floating-point computation at multiple levels of precision is common in ML training.

- **Packet loss.** Loss of packets is inevitable, so it’s necessary to devise a method of discovering and dealing with lost packets.

To deal with these challenges, the SwitchML prototype uses a combined switch-host architecture to appropriately divide the functionality between worker nodes and switches. The worker nodes drive the protocol, perform dynamic block-based quantization to convert floating-point values to fixed-point values, and address failures; the switch performs pool-based, fixed-point streaming aggregation, which helps mitigate the limited memory of networking switch ASICs. The design also includes an in-switch Remote Direct Memory Access (RDMA) implementation to achieve line rate processing even at 100 Gbps.

Each round of aggregation performs both fixed-point aggregation and the computation of the best scaling factors to use for the quantization of the next block of elements (see Figure 5). The scaling factors are chosen to maximize accuracy while avoiding overflow in the switch. Quantization allows the ML training to achieve nearly the same accuracy as a similar number of iterations on an unquantized network (see Figure 6).8

![Figure 5](image1.png)

**Figure 5.** Each round of aggregation performs both fixed-point aggregation and the computation of the best scaling factors to use for the quantization of the next block of elements.

![Figure 6](image2.png)

**Figure 6.** By using dynamic block-based aggregation, SwitchML technology can achieve nearly the same accuracy as an unquantized execution, even though the programmable switch ASICs do not include floating-point units (FPUs).8

The switch hardware is programmed with a P4 program for the Intel® Tofino™ Native Architecture (TNA) and managed at runtime through a Python controller using BF-Runtime. The host library that runs on the worker nodes provides simple APIs to perform Allreduce operations using different transport protocols. SwitchML currently supports user-space User Datagram Protocol (UDP) through the Data Plane Development Kit (DPDK) library and RDMA in Unreliable Connection (UC) mode. Resilience to packet losses is provided by the library, for any transport, through collaboration between worker nodes and the switch.

**Preliminary Results**

**Greater than 90 Percent Network Efficiency**

Given its limited computational resources, a switch can process only a certain number of elements. We implemented SwitchML on an Intel® Tofino™-based switch, which implements a Protocol Independent Switch Architecture (PISA). This switch ASIC has four pipelines to process up to 6.5 Tbps. We found that, in one pass, each pipeline can process 256 bytes of payload per packet. Given framing overhead, this results in a network efficiency (payload/packet size ratio) of 75.7 percent.

To further increase network efficiency, SwitchML implements a second strategy, based on chaining the pipelines together to process 1,024 bytes of payload per packet. This enables SwitchML to provide up to 92.6 percent network efficiency, which is the maximum efficiency possible with RDMA on Ethernet without using Jumbo Frames.
Up to 2.27x Speedup—Regardless of the Number of Workers

We measured the speedup that SwitchML provides when training different models on a testbed consisting of eight nodes and eight GPUs, compared to a baseline of NCCL using RDMA.

Figure 7 shows that SwitchML provides speedups between 1.13x to 2.27x, depending on the model (different models have different computation/communication ratios). These results were obtained on a 100 Gbps network using a single GPU per node and the UDP/DPDK transport. It is reasonable to expect that, using faster GPUs or multi-GPU nodes, the reduction in computation time would lead to even higher speedups for most models. Although we have not yet conducted additional tests, we expect to achieve even better performance with RDMA instead of UDP/DPDK, and we also expect speedups to be even larger for slower networks (such as 10 or 40 Gbps).

Building the Ecosystem

Intel is committed to advancing the Intel Tofino Intelligent Fabric Processor (IFP) ecosystem and pushing the boundaries of cloud-native technology to new heights. Intel is working with the ecosystem to develop SwitchML support for Intel® Tofino™ 2 and 3 IFP-based intelligent switches, which will provide up to 400 Gbps per port.

Switch to Intelligence

Introducing P4-programmable Intel® Tofino™ IFPs

Data centers are increasingly using software-defined infrastructure to improve efficiency and reduce costs. A software-defined network using programmable Intel® Tofino™ Intelligent Fabric Processors (Intel® Tofino™ IFPs) offers users customizable manageability of their packet processing, switching, and pipeline forwarding with intelligence, performance, visibility, and control—all optimized for data center fabric. Intel Tofino IFPs offer use-case-optimized power consumption, real-time in-band network telemetry, and enhanced congestion control that can span from the edge to the cloud. In particular, Intel Tofino IFPs can help provide intelligent packet processing to accelerate machine learning workloads using SwitchML technology. Network engineers can use Intel Tofino IFPs’ comprehensive suite of development and workload-monitoring tools to simplify integration of container-based microservices and pave the way to a more secure and self-healing infrastructure. With Intel Tofino IFPs, network engineers can spend more time advancing the network, instead of merely maintaining it.

Intel Is Leading the Way Toward Network-Based ML Acceleration

ML training workloads are increasingly becoming network-bound. While work remains to be done, SwitchML, currently available for prototyping, is well on its way to providing an easy-to-deploy Ethernet-based solution for eliminating communication-related bottlenecks during ML training. Compared to InfiniBand-based solutions, we expect SwitchML to be more affordable and immediately usable without having to upgrade infrastructure. Compared to a NCCL-RDMA solution, SwitchML provides up to a 2.27x speedup for model training.
Learn More

You may also find the following resources useful:

- Intel Tofino Intelligent Fabric Processors (IFPs)
- Intel Connectivity Academy
- Intel Connectivity Research Program
- SIGCOMM 2021 Tutorial: Network-Accelerated Distributed Deep Learning
- NSDI 2021 Presentation: Scaling Distributed Machine Learning with In-Network Aggregation

For more information, contact your Intel representative or visit the SwitchML Technology GitHub page.

Want to Be Part of the Solution?

Help the ecosystem co-innovate by sharing your ideas.

SwitchML technology continues to evolve. If you’d like to download and experiment with SwitchML, visit the SwitchML GitHub page. All SwitchML code is open source under the P4.org organization.

You can also get involved by joining the P4 programming community. Other related opportunities include projects at the Open Networking Foundation (ONF), like the recently launched P4 Integrated Network Stack (PINS).

---

1 OpenAI, “AI and Compute,” openai.com/blog/ai-and-compute/
4 For example, four Amazon EC2 t3.xlarge vCPU instances with 16.0 GiB memory, 40% baseline performance/vCPU, 5 Gbps network burst bandwidth, and up to 2.780 Mbps EBS burst bandwidth cost $0.1670/hr on-demand vs. 8 Amazon EC2 p4d.24xlarge GPU A100 instances (96 vCPUs) for ML training with 1152 GiB memory, 400 Gbps ENA and EFA network bandwidth, GPU direct RDMA, 300 GB/s GPU peer to peer NVSwitch, 8 x 1000 GB NVMe SSD instance storage, and 19 Gbps EBS bandwidth cost $32.77/hr on-demand. Prices shown are for Linux/Unix in US East (Northern Virginia) AWS Region. For full pricing details, see the Amazon EC2 pricing page. Source: Amazon EC2 T3 Instances and Amazon EC2 P4d Instances web pages, March 2022.
6 Ibid.
7 Ibid.
8 Ibid.
9 The network efficiency is calculated as a ratio between payload size and total packet size.
10 See endnote 5.