Serving DNNs in Real Time at Datacenter Scale with Project Brainwave

Eric Chung, Jeremy Fowers, Kalin Ovtcharov, Michael Papamichael, Adrian Caulfield, Todd Massengill, Ming Liu, Daniel Lo, Shlomi Alkalay, Michael Haselman, Maleen Abeydeera, Logan Adams, Hari Angepat, Christian Boehn, Derek Chiou, Oren Firestein, Alessandro Forin, Kang Su Gatlin, Mahdi Ghandi, Stephen Heil, Kyle Holohan, Ahmad El Husseini, Tamas Juhasz, Kara Kagi, Ratna K. Kovvuri, Sitaram Lanka, Friedel van Megen, Dima Mukhortov, Prerak Patel, Brandon Perez, Amanda Grace Rapsang, Steven K. Reinhardt, Bita Darvish Rouhani, Adam Sapek, Raja Seera, Sangeetha Shekar, Balaji Sridharan, Gabriel Weisz, Lisa Woods, Phillip Yi Xiao, Dan Zhang, Ritchie Zhao, and Doug Burger
Microsoft Corporation

To meet the computational demands required of deep learning, cloud operators are turning toward specialized hardware for improved efficiency and performance. Project Brainwave, Microsoft's principal infrastructure for AI serving in real time, accelerates deep neural network (DNN) inferencing in major services such as Bing's intelligent search features and Azure. Exploiting distributed model parallelism and pinning over low-latency hardware microservices, Project Brainwave serves state-of-the-art, pre-trained DNN models with high efficiencies at low batch sizes. A high-performance, precision-adaptable FPGA soft processor is at the heart of the system, achieving up to 39.5 TFLOPs of effective performance at Batch 1 on a state-of-the-art Intel Stratix 10 FPGA.

The recent successes of machine learning, enabled largely by the rise of DNNs, have fueled a growing demand for ubiquitous AI, ranging from conversational agents to object recognition to intelligent search. While state-of-the-art DNNs continue to deliver major breakthroughs in challenging AI domains such as computer vision and natural language processing, their computational demands have steadily outpaced the performance growth rate of standard CPUs. These trends have spurred a Cambrian explosion of specialized hardware, as large companies, startups, and research efforts shift *en masse* towards energy-efficient accelerators such as GPUs, FPGAs, and neural processing units (NPUs)[1-3] for AI workloads that demand performance beyond mainstream processors.

Solving the challenges of real-time AI is becoming increasingly important as cloud service providers deploy DNN-infused applications that ingest live data streams such as search queries, videos, sensor streams, and interactions with users. Bing's intelligent search features, for example, leverage state-of-the-art machine reading comprehension models to analyze and understand billions of documents from the web to provide more relevant answers faster and directly to users. However, real-time latency constraints severely limit the size, complexity, and quality of such models that can be deployed on conventional hardware at datacenter scale. While throughput-oriented architectures such as GPGPUs and batch-oriented NPUs are popular for offline training and serving, they are not efficient for online, low-latency serving of DNN models.

In this article, we describe Microsoft's principal platform for accelerated serving of DNNs, codenamed Project Brainwave. Project Brainwave is designed for real-time AI, which means the system can ingest a request as soon as it is received over the network at high throughput and at ultra-low latency without batching. Project Brainwave leverages the massive Intel FPGA infrastructure that Microsoft has been deploying over the last few years, which consists of a fabric of high-performance FPGAs attached directly to the datacenter network.[4-5] In this fabric, pools of FPGAs can be allocated as a shared hardware microservice callable by any CPU software on the shared network. Within a datacenter, up to O(100K) FPGAs can further communicate directly in as little as ten microseconds (one-way), reducing the friction by which workloads can be scaled across multiple FPGAs in a single microservice.

The Brainwave system leverages hardware microservices to parallelize and pin pre-trained DNN models across multiple FPGAs at scale. By pinning model parameters entirely in high-bandwidth on-chip memories, the FPGAs achieve near-peak processing efficiencies at low batch sizes, a critical requirement for real-time AI services. While many ASIC-based approaches developed by startups and research efforts also explore pinning of models in on-chip memory,[6] Brainwave's system architecture in an FPGA-rich datacenter-scale environment allows that capacity to scale with state-of-the-art DNN models in production.

The heart of the Project Brainwave system is a highly efficient soft NPU hosted on each FPGA that exposes an easy-to-use instruction set specialized for efficient serving of pre-trained DNN models. The microarchitecture of the Brainwave NPU is adaptable in both numerical precision and supported operators and scalable across three generations of Intel FPGAs. We show, for example, that an extremely large gated recurrent unit (GRU) model with five times the cost of ResNet-50 can be served in just under 1 millisecond (39.5 TFLOPs effective) on a single Stratix 10 280 FPGA.

The rest of this paper describes the Project Brainwave system in detail. We provide background on the system and tool chain that translates high-level models onto Brainwave NPUs. We then describe the soft NPU architecture, its microarchitecture, and implementation. Finally, we describe how Brainwave is used within Bing.

BACKGROUND

The Brainwave system targets Microsoft's hyperscale datacenter architecture with Catapult-enhanced servers, shown in Figure 1 (right), where server-attached FPGAs operate as first-class citizens on the datacenter network.[4] Each FPGA operates in-line between the server's network interface card (NIC) and the top-of-rack (TOR) switch, enabling in-situ processing of network packets and point-to-point connectivity between hundreds of thousands of FPGAs at low latency (two microseconds per switch hop, one-way). Catapult-enhanced servers have been deployed throughout Microsoft's commercial datacenters at hyperscale and are used for critical services such as accelerated software-defined networking and machine learning in Bing.[4-5]

While FPGAs are physically attached to Catapult-enhanced servers, their proximity to the network allows them to be disaggregated logically into CPU-independent resource pools exposed as hardware microservices (Figure 1, left). This disaggregation enables two critical capabilities: (1) the ability to reclaim underutilized resources for other services by rebalancing the subscription between CPUs and FPGAs, and (2) supporting workloads that cannot fit or run effectively on a

single FPGA. These capabilities are leveraged fundamentally in the deployment of Brainwave-based hardware microservices.

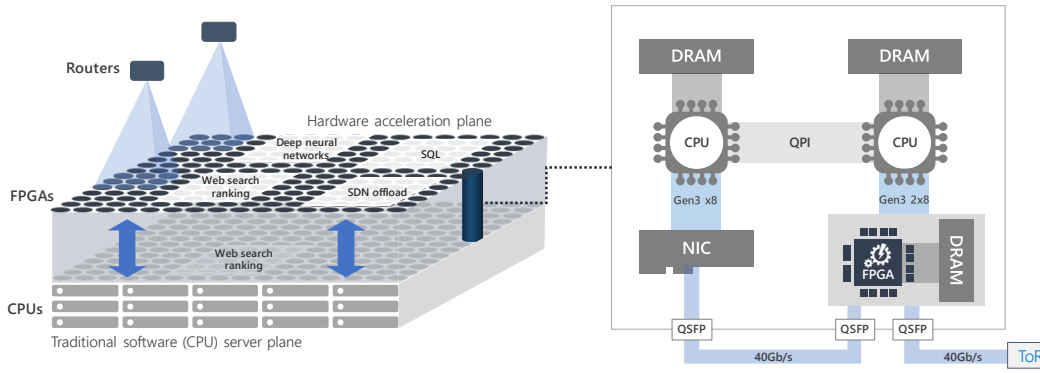

Figure 1. The first generation of Catapult-enhanced servers in production (right) consists of dual Xeon CPUs with a PCIe-attached FPGA. Each FPGA sits in-line between the 40Gbps server NIC and the TOR, enabling in-situ processing of network packets and point-to-point connectivity with up to hundreds of thousands of other FPGAs at datacenter scale. Multiple FPGAs can be allocated as a single shared hardware microservice with no software in the loop (left), enabling scalable workloads and better load balancing between CPUs and FPGAs.

BRAINWAVE ARCHITECTURE

The goal of Project Brainwave is to enable users without hardware expertise to automatically deploy and accelerate the serving of state-of-the-art DNN models in real time and at low cost. While convolutional neural networks (CNNs) have been studied widely and are relatively straightforward to accelerate, the pervasiveness of text-driven scenarios at Microsoft (web search, speech-to-text, neural machine translation, question-answer systems, etc.) has led to significant demand for state-of-the-art DNNs dominated by memory-intensive recurrent neural networks (e.g., long short-term memory units (LSTMs) and GRUs), attention layers, highway networks, 1D convolutions, and embeddings. These components of popular text-based models are much more bandwidth-intensive and more difficult to serve than CNNs, sustaining low hundreds to thousands of giga-ops per second at batch 1 on well-tuned CPU- and GPU-based implementations.

To meet real-time requirements on conventional hardware systems, such memory-intensive models must often be trimmed down in dimensionality and parameters, sacrificing quality and accuracy. To solve the challenges with serving memory-intensive DNNs, the Brainwave system exploits model parallelism and on-chip pinning at scale to achieve ultra-low latency serving of DNN models while preserving model accuracy. During offline compilation, the Brainwave tool flow splits DNN models into sub-graphs, each of which can fit into on-chip FPGA memory or run on the CPU. When sub-graphs are pinned to an FPGA, many terabytes/sec of memory bandwidth can be delivered to a single DNN query execution, enabling the FPGA to achieve ultra-low latency execution at near-peak efficiency—up to 90% utilization at batch 1.

There is an important synergy between on-chip pinning and the integration with hardware microservices. When a single FPGA's on-chip memory is exhausted, the system user can allocate more FPGAs for pinning the remaining parameters by scaling up a single hardware microservice and leveraging the elasticity of cloud-scale resources. This contrasts to single-chip NPUs (with no direct connectivity to other NPUs) designed to execute models stand-alone. Under these constraints, exploiting pinning would require scaling down models undesirably to fit into limited single-device on-chip memory or would require spilling of models into off-chip DRAM, increasing the single-batch serving latencies by an order of magnitude or more.

Project Brainwave Stack

Figure 2 illustrates the three main layers built to implement the Brainwave system:

1. a tool flow and runtime for low-friction deployment of trained models,
2. a distributed system architecture mapped onto CPUs and hardware microservices, and
3. a high-performance soft DNN processing unit synthesized onto FPGAs.

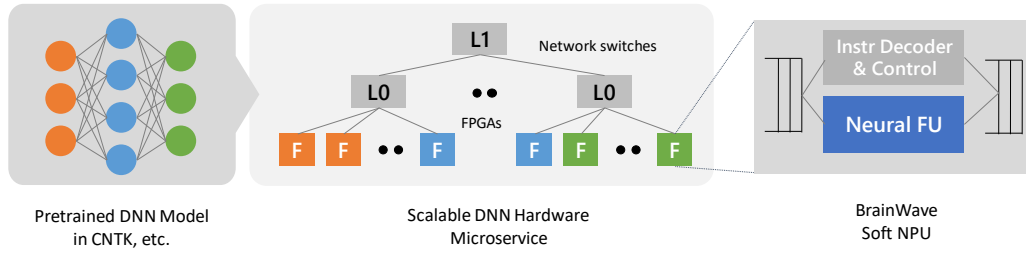

Figure 2. The three major layers of the Brainwave system: (1) a tool flow that converts pre-trained DNN models to a deployment package, (2) a scalable DNN hardware microservice with network-attached FPGAs, and (3) the programmable Brainwave NPU hosted on FPGA.

Figure 3 illustrates the tool flow that converts DNN models into a deployable hardware micro-service. A pre-trained DNN model developed in a framework of choice (e.g., Cognitive Toolkit (CNTK)[7] or TensorFlow) is first exported into a common graph intermediate representation (IR). Nodes in the IR represent tensor operations (e.g., matrix multiplication), while edges delineate the dataflow between operations.

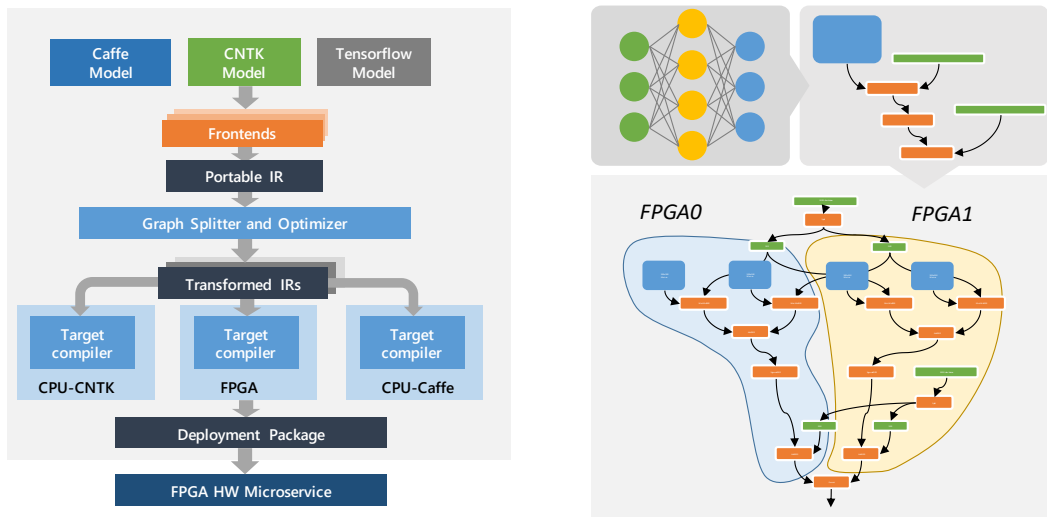

Figure 3. The framework-neutral Brainwave tool flow accepts models from different DNN toolchains and exports them into a common intermediate graph representation. Tool flow optimizes the intermediate representation and partitions it into sub-graphs assigned to different CPUs and FPGAs. Device-specific backends generate device assembly and are linked together by a federated runtime that gets deployed into a live FPGA hardware microservice.

The tool flow enables the user to partition the graph IR into different sub-graphs based on the constraints imposed by the target system, such as the type of layer (e.g. CNN vs. RNN), available number of FPGAs in a hardware microservice, the supported set of operators that can be accelerated on FPGA, and the available on-chip memory capacity per FPGA. Different approaches

are employed depending on layer type and degree of optimization. CNNs, due to their high computational intensity, are typically mapped to single FPGAs, while matrix weights of bandwidth-limited RNNs are pinned in a greedy fashion across multiple FPGAs during depth-first traversal of the DNN graph. Large matrices that cannot fit in a single device are sub-divided into smaller matrices. More sophisticated methods that factor in the cost of communication can be applied during optimization but are less critical since the datacenter network latencies we observe are small (a few microseconds) relative to compute times (a few milliseconds).

Operators that are unsupported or not profitable for offload are grouped into sub-graphs assigned to CPUs. This heterogeneous approach preserves operator coverage by leveraging CPUs as a catch-all for currently unsupported or rarely exercised operators (over time, however, nearly all performance-critical operators will be served on FPGAs). Users of the tool flow can also precisely control selection of sub-graphs and their assignments to specific resources using annotations in the model description. The second phase of the tool flow is shown in the bottom half of Figure 3, where partitioned sub-graphs are passed down to device-specific backend tools. Each backend may map sub-graphs into optimized libraries (e.g., AVX-tuned BLAS kernels on x86 CPUs) or compile optimized assembly automatically.

There are a multitude of ways to implement an FPGA backend. On one extreme, high-level synthesis can be used to synthesize custom circuits from high-level descriptions, but with poor efficiency. On the other extreme, hand-optimized RTL yields the most efficient implementations but requires high developer effort. To bridge the gap between productivity and efficiency, the Brainwave system targets a soft ISA. Instead of synthesizing soft logic from a neural network description directly, the FPGA backend tool targets a custom-built soft processor exposing a specialized instruction set for DNNs. The soft-processor architecture and microarchitecture are described further in the next section.

A federated runtime and scheduler consumes the device-specific executables for deployment. The federated runtime is responsible for integrating different runtime DLLs into a single executable (e.g., TF, CNTK, and FPGA host runtime), and executing sub-graphs scheduled to run on a specific runtime/device combination. The runtime also performs the marshalling of data between CPU/FPGA devices. The federated runtime and the backend binaries are packaged together and deployed to a live production service that allocates and deploys FPGAs to form a hardware microservice. The deployment process then publishes an access point to the hardware microservice to subscribing CPU clients in the system. Note that throughout the entire workflow, users require no FPGA knowledge or expertise to achieve accelerated serving.

When multiple FPGAs are allocated to host a single model, the aggregate FPGA capacity (e.g., sustainable queries per second) within a single hardware microservice tends to exceed the traffic that a single CPU client can drive. In practice, many CPU clients can be assigned to share an instance of a hardware microservice running on a set of FPGAs, better matching the compute ratio between CPU-FPGAs and avoiding the stranding of FPGA capacity.

BRAINWAVE NPU

The Brainwave NPU is a parameterized soft vector processor at the heart of the Brainwave system. Its salient features include:

(1) the use of compile-time narrow precision data types to extract higher performance than what is possible using conventional float and integer types without losses in accuracy;

(2) a simple, single-threaded programming model with an extensible ISA that can adapt to fast-changing DNN algorithms; and

(3) a scalable microarchitecture that maximizes hardware efficiency at low batch sizes.

These features enable the Brainwave NPU to close and exceed the gap between soft and hard NPUs by achieving better effective utilization (at low batch) and selecting more DNN-optimized data types beyond standard data types typical of GPGPUs and many hard NPUs.

The Brainwave NPU microarchitecture achieves high performance through “mega-SIMD’” execution, where a single issued instruction can produce over a million operations, sustaining up to

130,000 ops per cycle over 10 cycles (in the largest instantiation of the Brainwave NPU on an Intel Stratix 10 FPGA). Despite the massive spatial resources on the FPGA, the Brainwave NPU exposes a simple sequential programming model to users, employing dynamic control to extract parallelism transparently at all levels. This is enabled by a decoupled access/execute architecture with a sequential control processor that issues custom CISC instructions asynchronously to a decoupled neural functional unit (NFU), as shown in Figure 4.

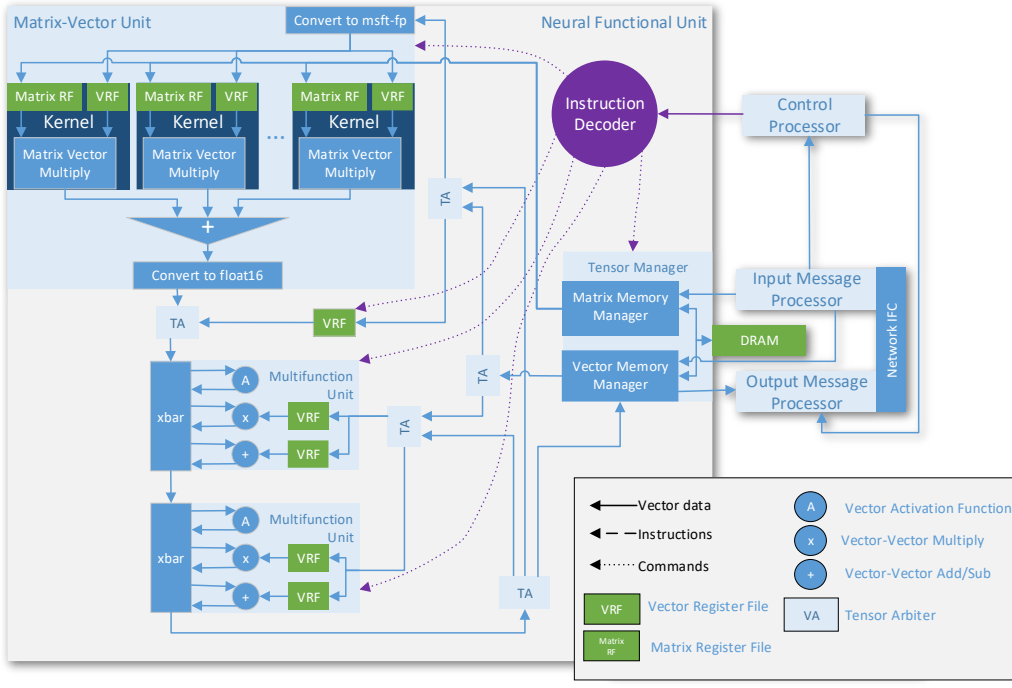

Figure 4. The Brainwave NPU is a "mega-SIMD" vector processor architecture. A sequentially programmed control processor asynchronously controls the neighboring neural functional unit (NFU) optimized for fast DNN operations. The heart of the NFU is a dense matrix vector multiplication unit (MVU) capable of processing single DNN requests at low batch with high utilization. The MVU is joined to secondary multifunctional units (MFU) that perform element-wise vector-vector operations and activation functions.

Hardware Organization

The control processor (shown in Figure 4 on the upper right) is implemented with an off-the-shelf Nios embedded processor. The Nios is initialized with offline-compiled firmware (developed in C or C++) and is responsible for issuing CISC instructions dynamically to the decoupled NFU through an asynchronous instruction queue. The NFU consists of specialized functional units for accelerating common-case DNN operations, such as dense matrix multiplication or activations. Nearby multi-ported vector register files provide temporary storage between operations. The Matrix-Vector Unit (MVU), the largest functional unit in the Brainwave NPU, receives encoded micro-ops from a distributed decoder that specifies op behavior (e.g., matrix size) and their dependences (e.g., source and target vector register). The MVU is further attached to secondary Multifunction Units (MFU) that implement activations and element-wise vector-vector operations on intermediate vectors.

The MVU consists of tens of thousands of parallel multiply accumulators (MAC) organized into parallel multi-lane vector dot product units (Figure 5). A salient feature of the MVU is the ability to maintain high utilization of all MAC resources when evaluating individual matrix-vector multiplication operations at a batch size of 1. Common implementations of DNNs rely heavily on dense matrix multiplication, where weight matrix columns encode the pre-trained synaptic

weights of neurons, while rows of the activation matrix represent independent inputs to the neural layer.

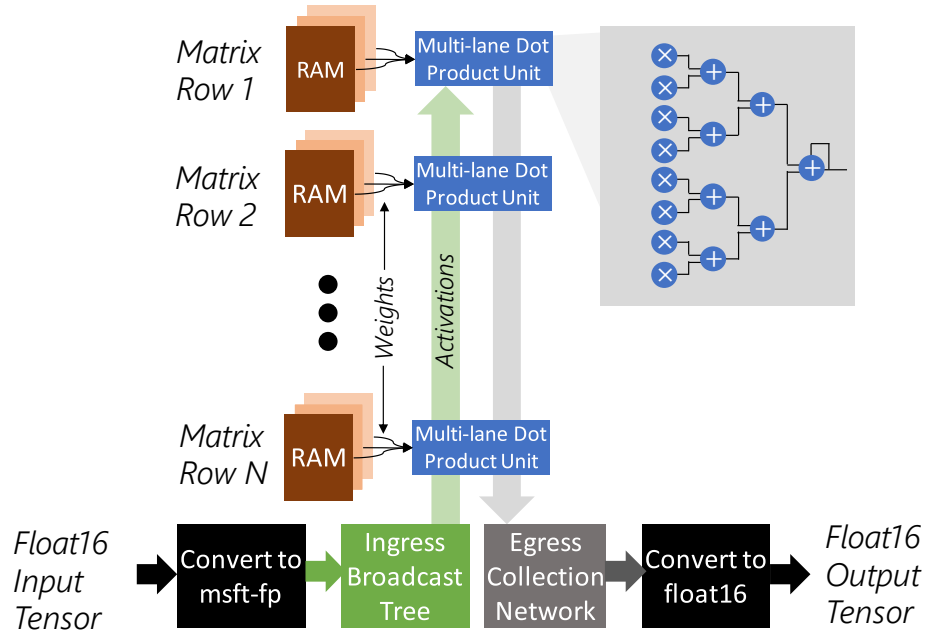

Figure 5. An independent SRAM memory port is dedicated to every lane of a multi-lane vector dot product unit within the MVU, allowing up to 80,000 MACs on a Stratix 10 280 to be fed with independent weights. As a result, FPGA can achieve near peak utilization on batch 1-oriented matrix-vector multiplication.

In the case of batch-1 serving for real-time processing, the activation matrix is sized to a single row, limiting weight reuse to 1. This operation is inherently memory-bandwidth limited, making it difficult for parallel architectures to extract full utilization of spatial compute resources. Resizing the activation matrix to multiple rows increases the batch size and drives up hardware utilization (a common tactic used to achieve higher flops in GPUs and batch-oriented NPUs), but does not improve the performance and, hence, latency of a single DNN request.

Applying the system-level strategy of on-chip pinning discussed previously, the Brainwave NPU leverages the abundant independent on-chip memory resources of FPGAs to pin and serve DNN weight parameters to MAC units (without reload between independent DNN requests to the NPU). A contemporary Intel Stratix 10 280 FPGA, for example, has 11,721 independently addressable 512x40b SRAMs, and, in aggregate, provides 30 MB of on-chip storage and 35 terabytes/sec of bandwidth at 600 MHz. Typical instances of Brainwave NPUs on Stratix 10 allocate storage for tens of millions of parameters and hundreds of thousands of vector elements. While pinning is the default strategy for most NNs, DRAM can also be used for buffering activations and weights.

As shown in Figure 5, multi-lane vector dot product units form the bulk of the MVU. Locally attached SRAMs feed each MAC with unique weight values, enabling full-chip utilization of memory-intensive matrix-vector multiplication operations. Although the MACs operate in lower precision, their internal data types are transparent to the overall NPU and software (via *in-situ* hardware converters), which operate in float16 for I/O and non-dot product operations.

Narrow Precision

State-of-the-art deep neural networks are tolerant to noise and can maintain acceptable levels of accuracy even when weights and activations are expressed in low numerical precisions. While

modern GPUs and hard NPUs for inference offer reduced precisions as low as 8-bit integer, reducing numerical precision even further is possible without losses in accuracy. Based on accuracy-sensitivity studies of internal production and public DNN models (e.g., ResNet), we have developed proprietary "neural"-optimized data formats based on 8- and 9-bit floating point, where mantissas are trimmed to 2 or 3 bits. These formats, referred to as ms-fp8 and ms-fp9, exploit efficient packing into reconfigurable resources and are comparable in FPGA area to low bit-width integer operations but can achieve higher dynamic range and accuracy than pure fixed-point implementations. We have found that the versatility of these formats simplifies the quantization process of neural networks considerably relative to conventional 8-bit fixed point, where significant fine-tuning in the radix point is needed to preserve accuracy.

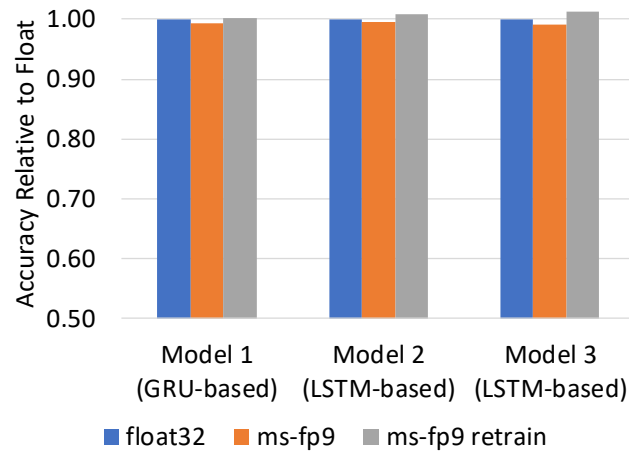

Figure 6. The impact to accuracy of various DNN production models is negligible with post-trained quantization into narrow precision ms-fp. With re-training, accuracy can be fully recovered.

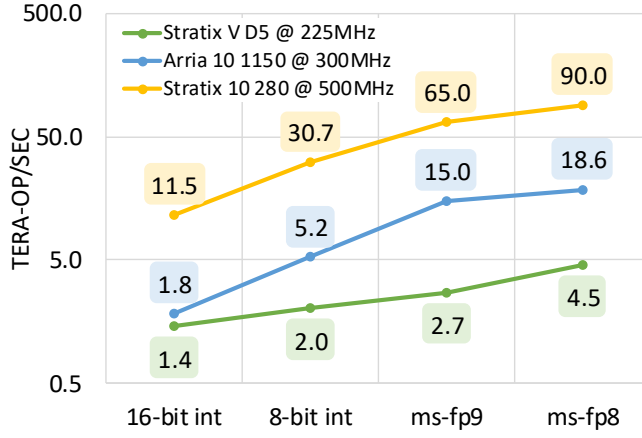

Figure 7. NPU Peak performance of the Brainwave DPU across three generations of Intel FPGAs. The use of ms-fp8 narrow precision improves performance by 3.2X-7.8X over a conventional 16-bit fixed point.

Figure 6 shows three internal production models that quantize effectively to ms-fp9. A non-retrained conversion results in a minimal degradation in accuracy (1-3%), which can be fully recovered with just 1-3 epochs of quantized retraining. When optimized for ms-fp8 and ms-fp9 arithmetic, the Brainwave NPU can achieve high levels of MAC density and peak performance. Figure 7 compares peak TFLOPs when precision is varied, showing up to 90 TFLOPs of peak

performance on a Stratix 10 280 at ms-fp8. Note that these levels of performance are comparable to high-end hard NPUs and contemporary GPUs implemented with higher-precision fixed-point arithmetic.

RESULTS

Accelerating Bing Intelligent Search

As an early pioneer of FPGAs in the datacenter, Bing now routinely utilizes FPGAs for accelerated feature extraction, serving of DNN models, and other expensive computations. The Bing environment is particularly challenging, with real-time latency requirements that can thwart the practical deployment of unconstrained state-of-the-art DNNs on CPUs. Furthermore, the algorithms and models used within Bing evolve at a rapid pace with new models experimented in production each day. In response, Microsoft engineers regularly adapt the Brainwave NPU ISA and microarchitecture to new requirements, pushing modified FPGA images into production environments at a cadence from days to weeks.

Bing's intelligent search features leverage the Turing Prototype 1 (TP1) and DeepScan DNN models to identify a relevant set of passages for a given query from billions of documents, from which the best answer for the query can be derived. Bing uses these DNNs to process data from multiple reputable sources, enabling multiple perspectives or collective knowledge in cases when the search question has more than one answer. If different authoritative perspectives from reputable sources exist for a given topic (for instance, benefits versus drawbacks), Bing will aggregate the two viewpoints and surface them toward the top of the search page, saving time for the user.

Table 1 compares the end-to-end latencies of TP1 and DeepScan deployed in Bing's production datacenters using CPU vs. CPU+Brainwave (including software overheads). To meet end-to-end latency requirements, the CPU-only versions are scaled down in both parameters and ops by an order of magnitude relative to Brainwave-accelerated versions. The scaled down TP1 incurs 9 ms (95^{th} percentile) on Bing's CPU-only infrastructure. Brainwave, on the other hand, enables serving of a larger, more accurate bidirectional LSTM variant of the model, with hidden states increased from 128 to 500. The model is > 10X the size of the CPU-only model and > 10X lower latency at 0.85 ms (95^{th} percentile). Thus, the space-time speedup Brainwave provides is more than 100X. The CPU-only DeepScan further omits RNN computation entirely while still incurring 3X the tail latency of Brainwave.

Table 1. Comparison of CPU-only vs. Brainwave-accelerated TP1 and DeepScan DNN models in Bing production.

Bing TP1			
	CPU-only	**Brainwave-accelerated**	**Improvement**
Model details	GRU 128x200 (x2) + W2Vec	LSTM 500x200 (x8) + W2Vec	Brainwave-accelerated model is > 10X larger and > 10X lower latency
End-to-end latency per Batch 1 request at 95%	9 ms	0.850 ms	
Bing DeepScan			
	CPU-only	**Brainwave-accelerated**	**Improvement**
Model details	1D CNN + W2Vec *(RNNs removed)*	1D CNN + W2Vec + GRU 500x500 (x4)	Brainwave-accelerated model is > 10X larger and 3X lower latency
End-to-end latency per Batch 1 request at 95%	15 ms	5 ms	

Brainwave NPU on Pre-production Stratix 10 280

The Brainwave NPU on a more contemporary FPGA substantially increases the size of models that can be served in real time at datacenter scale. In this section, we report on an instance of the Brainwave NPU scaled up to run on a pre-production Intel Stratix 10 280 FPGA. The Stratix 10 architecture incorporates significant improvements over previous FPGA generations, including pipelined interconnect registers that enable improved clock speeds (600 MHz or higher), higher density of DSPs and memories, and improved power and delay from Intel's 14-nm process. Compared to the Stratix V D5 FPGA, the Stratix 10 280 has 3X higher clock speed, 5.4X more soft logic resources, 3.6X more DSPs, and 5.8X additional on-chip memory.

These resources enabled instantiation of a Brainwave NPU with 80,000 ms-fp8 MACs operating at 300 MHz on pre-production silicon (48 TFLOPs peak). In end-to-end measurements, including host overheads, this engine can serve a Batch-1 high-dimensional 3,200x2,400 GRU with 480 timesteps in under 1 ms. To put this measurement in perspective, the 39 billion operations in this benchmark is 5X that of ResNet-50 and about 25X that of AlexNet. The effective throughput measured was 39.5 TFLOPs out of 48 TFLOPs peak, operating at 82% efficiency at batch 1. With production tools and silicon, the design should comfortably reach 550 MHz, for an additional 83% gain. In terms of energy efficiency and power, the Stratix 10 consumes 125 W when measured with a fully loaded power virus. On production silicon, the expected energy efficiency at peak throughput is 720 GOPs/watt.

CONCLUSION

Hastened by the escalating demand for deep learning, the march toward ubiquitous specialized hardware for AI is well underway. There are many approaches being pioneered by companies, startups, and research efforts—spanning GPGPUs to NPUs. Project Brainwave, Microsoft's principal infrastructure for accelerated AI serving in the cloud, successfully exploits FPGAs on a datacenter-scale fabric for real-time serving of state-of-the-art DNNs.

The key learnings of Project Brainwave are: (1) designing a scalable, end-to-end system architecture for deep learning is as critical as optimizing for single chip performance—in Brainwave, the system and the soft NPU are co-architected in mind for each other, exploiting datacenter-scale pinning of models in on-chip memories that scale elastically beyond single-chip solutions, (2) narrow precision quantization is a viable approach for production DNN models, enabling the Project Brainwave system to achieve competitive levels of performance and energy efficiency (720 GOPs/W on an Intel Stratix 10 280) to hard NPUs with standard precisions without degrading accuracy, and (3) using configurable hardware at scale, a system can be designed without an adversarial tradeoff between latency and throughput (batching)—Brainwave is able to serve models at ultra-low latency at Batch 1 without compromising on throughput and efficiency.

In the future, real-time AI will become increasingly adaptive to live data, necessitating converged architectures for both low-latency inferencing and high-throughput training. State-of-the-art deep-learning algorithms are also evolving at a blinding pace, requiring continuous innovation of the Brainwave architecture. Today, Project Brainwave serves DNNs in real time for production services such as Bing and Azure and will become available to customers in 2018.

ACKNOWLEDGEMENTS

Many collaborators and partners contributed to the research, support, deployment, and uses of Project Brainwave. We thank Mike Andrewartha, Anthony Aue, Gregg Baeckler, Jing Bai, Jeff Baxter, Sebastian Blohm, Ted Briggs, Luis Castillo, Qingzheng Chen, Steve Clayton, Jacob Devlin, Yuanyuan Ding, Pavel Dournov, Mario Drumond, Alexander Gaunt, Atul Gupta, Daniel Hartl, Matt Humphrey, Alok Jain, Matthew Johnson, Max Kalinin, Qifa Ke, Caleb Kierum, Aaron Landy, Martin Langhammer, Jack Lavier, Zhuowei Li, David Liu, Robert Kirchgessner, Michael Lazos, Jingwen Lu, Rangan Majumder, Arul Menezes,

Oana Nicolov, James Olson, Kaan Ozel, Jongse Park, Maharshi Patel, Anjana Parthasarathy, Alex Prodan, Andrew Putnam, Saravanakumar Rajmohan, Ransom Richardson, Srikanth Shoroff, Ashvin Supekar, Mir Rosenberg, Victor Rühle, Guenther Schmuelling, David Shih, Jamie Shotton, Xia Song, Shawn Swilley, Daniel Tarlow, Blaise Tine, Saurabh Tiwary, Ryota Tomioka, Raja Venugopal, Colin Versteeg, Anya Vinogradsky, Dimitrios Vytiniotis, Tong Wang, Yestin Wang, Weihu Wang, Ted Way, Sam Webster, Alex Wetmore, Yiping Zhou, and many others. We are especially grateful for the strong executive support from David Ku, Peter Lee, Richard Qian, Jordi Ribas, Harry Shum, and Joseph Sirosh. We thank Intel for their excellent partnership and support for Project Brainwave.

REFERENCES

1. J. Ouyang et al., “SDA: Software-defined accelerator for large-scale DNN systems,” *IEEE Hot Chips 26 Symposium* (HCS), 2014; http://ieeexplore.ieee.org/document/7478821/.
2. K. Guo et al., “From model to FPGA: Software-hardware co-design for efficient neural network acceleration,” *IEEE Hot Chips 28 Symposium* (HCS), 2016; http://ieeexplore.ieee.org/document/7936208/.
3. N. Jouppi et al., “In-Datacenter Performance Analysis of a Tensor Processing Unit,” *Proceedings of the 44th Annual International Symposium on Computer Architecture* (ISCA), 2017; www.computer.org/csdl/proceedings/isca/2017/4892/00/08192463-abs.html.
4. Bing Intelligent Search. https://www.bing.com/explore/intelligentsearch.
5. A. Putnam et al., “A reconfigurable fabric for accelerating large-scale datacenter services,” *Proceeding of the 41st Annual International Symposium on Computer Architecture* (ISCA), 2014; http://ieeexplore.ieee.org/document/6853195/.
6. A. Caulfield et al., “A Cloud-Scale Acceleration Architecture,” *49th Annual IEEE/ACM International Symposium on Microarchitecture* (MICRO), 2016; www.computer.org/csdl/proceedings/micro/2016/3508/00/07783710-abs.html.
7. Y. Chen et al., “DaDianNao: A Machine-Learning Supercomputer,” *47th Annual IEEE/ACM International Symposium on Microarchitecture* (MICRO), 2014; http://ieeexplore.ieee.org/document/7011421/.
8. F. Seide and A. Agarwal, “CNTK: Microsoft’s Open-Source Deep-Learning Toolkit,” *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD), 2016; https://dl.acm.org/citation.cfm?id=2945397.
9. M. Abadi et al., “TensorFlow: A system for large-scale machine learning,” *12th USENIX Symposium on Operating Systems Design and Implementation* (OSDI), 2016; https://dl.acm.org/citation.cfm?id=3026899.