AI INFERENCING WITH AMD EPYC™ PROCESSORS

AMD
together we advance_AI

March 2023

AI INFERENCING WITH AMD EPYC™ PROCESSORS

CONTENTS

AI IS PERVASIVE

Everywhere you look, artificial intelligence powers our business world. Once an aspirational endeavor, vast leaps in computer power have turned hopes into reality. Information Technology (IT) organizations recognize how ubiquitous AI has become and how easily their businesses can be left behind if they don't begin to use it to innovate in their business. This is why AI is used across areas including commercial and enterprise, cloud data centers, transportation, smart retail, healthcare and life sciences, smart homes, intelligent factories, smart cities, and communication providers.

AMD PROPELS THE AI LIFECYCLE

The two most important parts of the deep learning lifecycle are AI training and AI inferencing:

- **AI TRAINING.** This is the most data- and processing-intensive part of the AI lifecycle. Vast amounts of data are fed through models in order to train them to recognize patterns in data. The large amount of processing needed to train a model requires a significant amount of computing power, and servers equipped with AMD Instinct™ accelerators are designed to accelerate the process. The goal of this part of the AI lifecyle is to arrive at the weights of parameters to make the model function as desired with the required accuracy.
- **AI INFERENCING.** Once the model is trained, it needs a comparatively small amount of processing power to process incoming data in real time. While models are trained at the beginning of the process and need to have a large amount of concentrated power, inferencing happens close to the data: in a retail store, in a moving automobile, on factory floors, in radiology departments. The locus of where computing power meets data is the point where efficiency is everything.

Embedded inferencing devices are used in applications such as surgical robotics, security endpoints, and smart-grid applications. These often use specialized hardware that incorporates highly flexible devices such as AMD Versal™ and Zynq™ adaptive adaptive systems on chip (SoCs).

INFERENCING WITH OFF-THE-SHELF TECHNOLOGY

A highly economical solution is to use off-the-shelf servers equipped with AMD EPYC™ processors. While training is a highly compute-intensive operation, inferencing uses the parameters from training to actually execute the model, which imposes much lower compute demands than training. With up to 96 cores per 4th Gen AMD EPYC processor, off-the-shelf servers can accelerate a range of data center and edge applications including customer support, retail, automotive, financial services, medical, and manufacturing. These models typically fall into the categories of computer vision, natural language processing, and recommendation systems.

Our approach, using a unified inferencing model, facilitates training on high-performance GPU hardware, and then enables you to seamlessly move the model into production with inferencing on servers with AMD EPYC processors. We offer software support throughout your AI lifecycle, inferencing in the core, cloud, and edge, and access to optimized models and software stacks that complement AMD hardware.

BROAD INDUSTRY IMPACT

Three different types of models have had a dramatic impact on business across multiple industries. Computer vision can recognize and classify objects, and also detect anomalies. Natural language processing can help recognize speech and make meaning out of written words to assist customers of all types. Recommendation systems can help predict everything from customer needs to anomalies in telemetry data. By focusing on accelerating these three model classes, you can reap the benefits regardless of your industry.

Automotive
Computer vision models help propel self-driving cars and also help recognize signage, pedestrians, and other vehicles to be avoided. Natural-language processing models can help recognize spoken commands to in-car telematics.

Manufacturing
Use computer vision models to monitor quality of manufactured products from food items to printed-circuit boards. Feed telemetry data into recommendation engines to suggest proactive maintenance: Are disk drives about to fail? Is the engine using too much oil?

Retail
Automate checkout lines by recognizing products, or even create autonomous shopping experiences where the models link customers with the items they choose and put into their bags. Use product recommendation engines to offer alternatives, whether online or in the store.

Financial Services
AI-powered anomaly detection helps stop credit-card fraud, while computer vision models watch for suspicious documents including customer checks.

Medical
Detect anomalies including fractures and tumors with computer vision models. Use the same models in research to assess in vitro cell growth and proliferation.

Service Automation
Where IT meets customers, natural-language processing can help take action based on spoken requests, and recommendation engines can help point customers to satisfactory solutions and product alternatives.

UNIFIED INFERENCING MODEL

Putting AI models to work across industries means that developers must be able to move seamlessly from training to inferencing and for those operations to run with high performance. To this end, we support three AI software stacks, one for each of our three architectures, including AMD EPYC processors, AMD Instinct GPUs, and Xilinx Versal and Zynq adaptive SoCs (see grey components in Figure 1). Each of these three stacks are optimized to deliver excellent performance on the underlying hardware.

The AMD Unified Inference Frontend (UIF), which is represented by the blue components in Figure 1, provide consistent access to these stacks through the three most popular frameworks for AI. Underneath each framework are tools and libraries optimized for the supporting hardware platform. The frameworks we support include:

- **TENSORFLOW:** This Google-owned platform focuses on training and inference of deep neural networks.
- **PYTORCH:** Originally developed by Meta, PyTorch was recently welcomed into the Linux® Foundation.
- **ONNX RUNTIME:** The Open Neural Network Exchange (ONNX), a Microsoft-sponsored platform.

AMD UNIFIED INFERENCE FRONTEND

In order to provide a single entry point for model development and deployment, UIF provides a set of optimized inference models in a single model zoo. These optimized models plug seamlessly into each of three hardware stacks and, because they are transportable, your model can run on any of our stacks without modification. Now you gain the power to use the best hardware platform for AI inferencing and the flexibility to change platforms as different needs arise. As we see customers mixing CPUs, GPUs, and adaptive SoCs in their inferencing operations, the Unified Inference Frontend can enhance the AI lifecycle. A few of the currently available optimized models are presented in Table 1. Both the models and UIF are available on GitHub at https://github.com/AMD/UIF.

FLEXIBILITY PLUS PERFORMANCE

With UIF, you gain not just flexibility, but also performance. The UIF taps into the performance-enhanced versions of each of today's software stacks and draws upon the ZenDNN library for AMD EPYC processors, the open AMD ROCm™ stack for AMD Instinct,

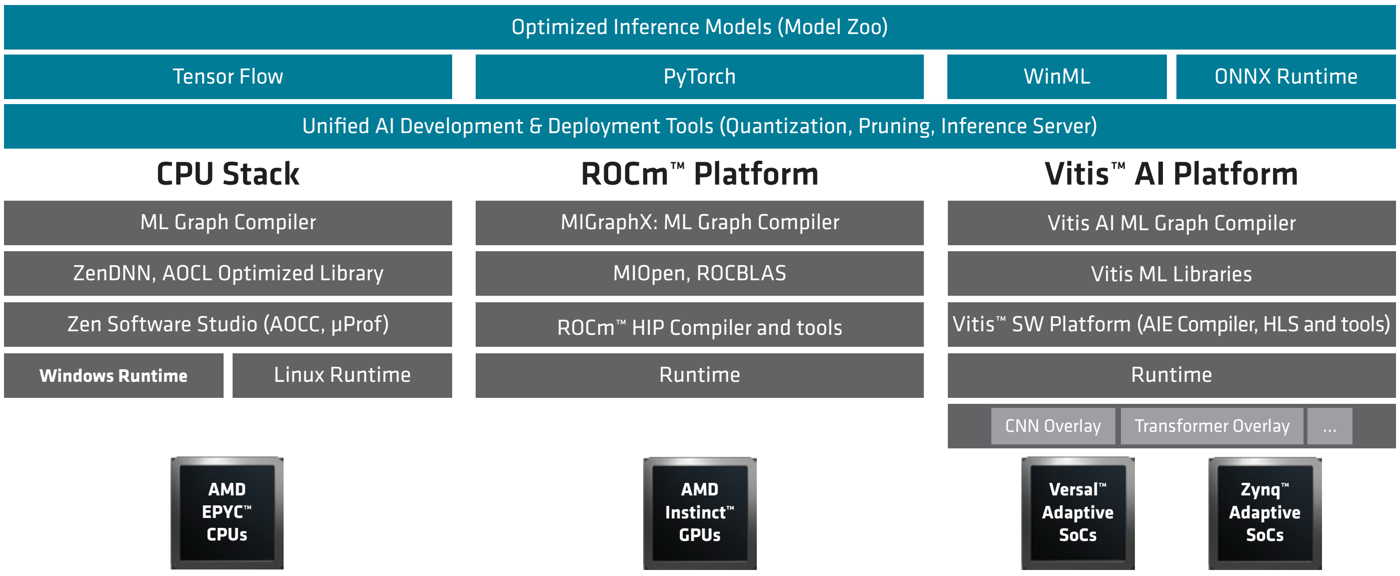

Optimized Inference Models (Model Zoo)
Tensor Flow
PyTorch
WinML
ONNX Runtime
Unified AI Development & Deployment Tools (Quantization, Pruning, Inference Server)
CPU Stack
ROCm™ Platform
Vitis™ AI Platform
ML Graph Compiler
ZenDNN, AOCL Optimized Library
Zen Software Studio (AOCC, µProf)
Windows Runtime
Linux Runtime
MIGraphX: ML Graph Compiler
MIOpen, ROCBLAS
ROCm™ HIP Compiler and tools
Runtime
Vitis AI ML Graph Compiler
Vitis ML Libraries
Vitis™ SW Platform (AIE Compiler, HLS and tools)
Runtime
CNN Overlay
Transformer Overlay
...
AMD EPYC™ CPUs
AMD Instinct™ GPUs
Versal™ Adaptive SoCs
Zynq™ Adaptive SoCs

Figure 1: The Unified Inference Frontend (blue) provides a uniform way to link your inferencing software with the acceleration capabilities of our EPYC CPUs, Instinct accelerators, and Versal and Zynq adaptive SoCs.

and the Vitis™ AI platform for Versal and Zynq adaptive SoCs. The optimizations made for the three hardware platforms are automatically incorporated when you deploy inferencing operations on UIF.

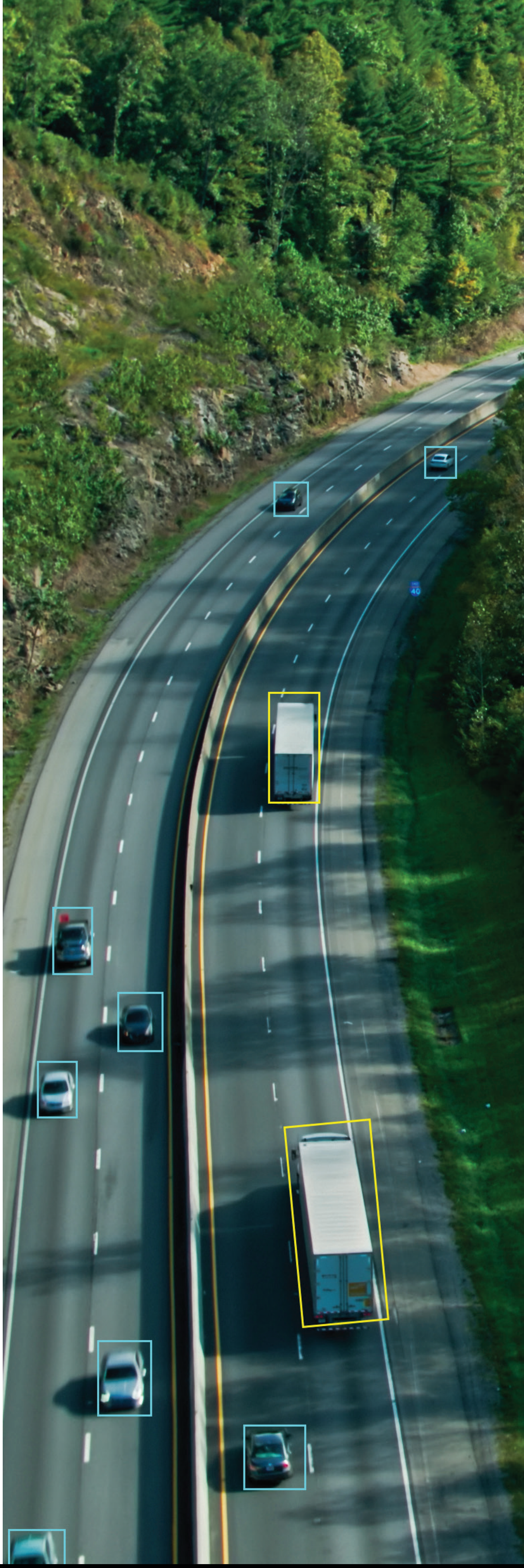
The underlying libraries optimize performance for the hardware, and the models in the zoo are optimized themselves to deliver exceptional performance right out of the box. For example, consider the Medical EDD RefineDet model from the AMD UIF 1.1 model zoo. The baseline model uses FP32 data types and quantizes them to INT8, reducing processing time by addressing higher throughput needs without upgrading compute resources. Optimizing the model pruned 88% of the computation, and it used the ZenDNN library, further reducing resource needs. Testing this model on a server with 2x 96-core AMD EPYC 9004 Series engineering samples delivered up to 20.50x the images/second throughput compared to the baseline FP32 model from the same model zoo.[ZD-036]

Both the UIF 1.0 and 1.1 models are available from GitHub at: https://github.com/amd/UIF/tree/main/docs/2_model_setup/model-list. More than 100 models are available to run on the ZenDNN library, with more in process. Example classes of models are listed in Table 1. If you view the list of models by standardized naming conventions, those with the Z4.0 suffix are configured to run on ZenDNN 4.0.

The optimized libraries and the optimized model zoo deliver compounded performance increases that help propel your inferencing operations to new heights. The focus of the remainder of this white paper is the advancement to our inferencing libraries for use with AMD EPYC processors.

Table 1: Example models supported by UIF

MODEL	FRAMEWORK	DATA TYPES	AVAILABLE PRUNING
ResNet-50	TensorFlow[ZD-007]	FP32, INT8	38%, 65%
	PyTorch[ZD-011]	FP32, INT8	30%, 40%, 50%, 60%, 70%
VGG16	TensorFlow[ZD-010]	FP32, INT8	43%, 50%
Inception v3	TensorFlow[ZD-009]	FP32, INT8	20%, 40%
	PyTorch[ZD-0012]	FP32, INT8	30%, 40%, 50%, 60%



INFERENCING WITH AMD EPYC PROCESSORS

AI inferencing generally takes place close to the data, and servers with AMD EPYC processors are often there too, ready to take on the task. In retail environments, video streams can be processed for monitoring inventory with on-site edge servers. In manufacturing, assembly-line images of products can be inspected for defects. In medical imaging applications, hospitals already store and use images on their centralized servers. Financial and consumer services are generally centralized, so the data that needs to be analyzed is already collocated with servers often powered by AMD EPYC processors.

Whether at the core or at the edge, using servers with AMD EPYC processors for inferencing is an easy choice. Our 4th Gen AMD EPYC processors have made huge gains in areas that accelerate inferencing operations, enabling a hardware boost. Our Unified Inference Frontend includes software optimizations that help make models running with our software run at peak performance.

AMD EPYC PROCESSOR ADVANCEMENTS

AMD EPYC processors power the servers most often selected for AI inferencing, making them the AI platform of choice.[EPYC-037] Now, with 4th Gen EPYC processors, AMD is making huge strides in further improving the aptitude of AMD EPYC processors for performing AI inferencing, including the following:

- **'ZEN 4' CORE:** Optimizations in the core architecture have resulted in an overall ~14% increase in instructions per clock cycle compared to our 3rd Gen processors. This improvement is based on the geometric mean of 33 integer, floating-point, and other server workloads.[EPYC-038]
- **MORE CORES:** With a 50% increase in cores from a maximum of 64 in the prior generation to 96 in the current generation, more parallelism is available to power inferencing operations without GPU acceleration.
- **AVX-512 INSTRUCTION EXTENSION SUPPORT:** The 'Zen 4' core now supports AVX-512 extensions. The VNNI component enables significant gains in AI inferencing performance across a range of data types. AVX-512 supports BF16 data types for improved throughput without having to risk the quantization challenges of using INT8 data. The AVX-512 implementation uses a two-cycle, efficient 256-bit pipeline and thus could help maintain frequencies higher than with AVX-2. (Actual achievable frequency will vary depending on hardware, software, workloads, and other conditions.)[EPYC-039]
- **FASTER MEMORY WITH MORE CHANNELS:** The combination of DDR5 memory, plus 50% more memory channels (total of 12) in 4th Gen AMD EPYC CPUs yields a total of 2.25x the memory throughput compared to our prior generation.[EPYC-040]
- **FASTER I/O:** PCIe® Gen 5 I/O, again doubles AMD EPYC processor I/O throughput over the prior generation, enabling faster data ingest for AI inferencing.

PROCESSOR PERFORMANCE PROOF

Performance measurements by AMD and third parties confirm the benefits of using AMD EPYC processors for inferencing. Enabling AVX-512 instructions makes image processing run dramatically faster while also increasing the amount of work done per watt—proof of the efficiency of our implementation. Servers with 2x AMD EPYC 9654 processors can recognize vehicles 78% faster than servers with 2x 4th Gen Intel® Xeon® Scalable 8490H processors according to measurements by Phoronix on the OpenVINO Vehicle Detection FP16-INT8 model. Comparing our own server processors generation over generation, all three of the areas in which we strive to excel demonstrate dramatic speedups: computer vision, natural language processing, and recommendation engines. For details on all of these measurements, see the next page.

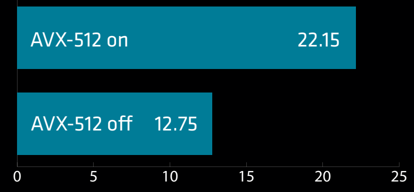
The bottom line is that when you are considering using server CPUs for AI inferencing, AMD EPYC processors can deliver the performance you need with superb energy efficiency characteristics. And those are just the hardware measurements. Next, we add the benefits of our software optimizations.

AMD EPYC PROCESSOR PERFORMANCE PROOF

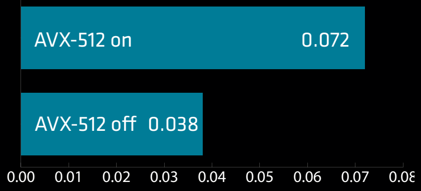
UP TO **1.73x**

OUR AVX-512 INSTRUCTION DELIVERS A 1.73X PERFORMANCE SPEEDUP WHILE ALSO INCREASING THE AMOUNT OF WORK DONE PER WATT

ResNet-50 BF16 Images per second
Higher is better



ResNet-50 BF16 images per second per watt
More is better

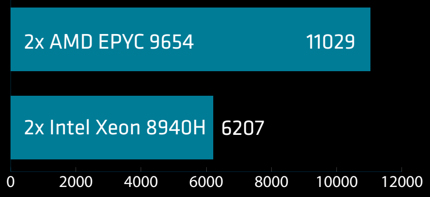


Phoronix has measured the performance impact of the AMD EPYC 9654 processor's AVX-512 instruction by running the Res-Net-50 model with TensorFlow 2.10 and the BF16 data type. They ran the benchmark on the same server with AVX-512 turned on vs. off. They measured a 1.73x performance improvement with AVX-512 turned on, a minimal impact on clock frequency, and almost double the images processed per second per watt. This data demonstrates the efficiency of our AVX-512 implementation.

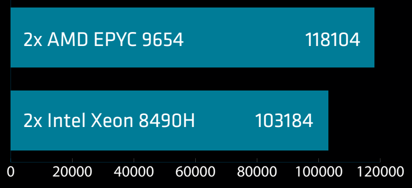
UP TO **1.78x**

SERVERS BASED ON AMD EPYC PROCESSORS RECOGNIZE VEHICLES AT 1.78X THE RATE OF INTEL XEON PROCESSOR-BASED SERVERS.

OpenVino FP16-INT8 vehicle detection
FPS, Higher is Better



OpenVINO FP16-INT8 age gender recognition
Faces per second, higher is better

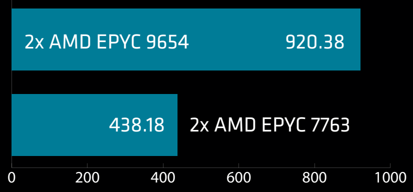


Phoronix used the OpenVINO benchmark using INT8-FP16 data types to compare multiple CPU types. They measured a whopping 78% speedup on vehicle detection, and a 14% speedup on age-gender recognition comparing a 2P Intel Xeon 8490H processor-powered server to a 2P AMD EPYC 9654 processor-powered server.

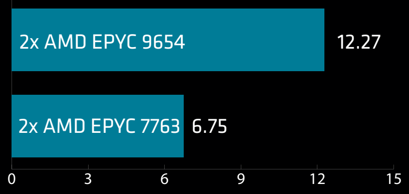
UP TO **2.10x**

GENERATION OVER GENERATION, THE EPYC 9004 SERIES DEMONSTRATES SPEEDUPS OVER THE 7003 SERIES IN ALL THREE AREAS

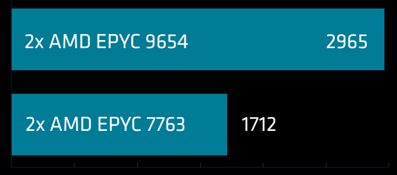
COMPUTER VISION–2.10X[ZD-037]
ResNet50 INT8 images/sec, more is better
batch size=640



NATURAL LANGUAGE PROCESSING[ZD-031]
BERT-Large samples/sec, more is better
batch size=32



RECOMMENDATION ENGINE[ZD-038]
DLRM queries/sec, more is better
batch size=1



SOFTWARE OPTIMIZATIONS

With the Unified Inferencing Frontend, you can pick the deep learning model you want to deploy, and execute your inferencing on the framework of your choice, including PyTorch, TensorFlow, and ONNX Runtime. Once you've made that decision, you can run your inferencing model on any of our three supported hardware platforms.

The CPU-specific software stack (left side of Figure 1) includes the following tools:

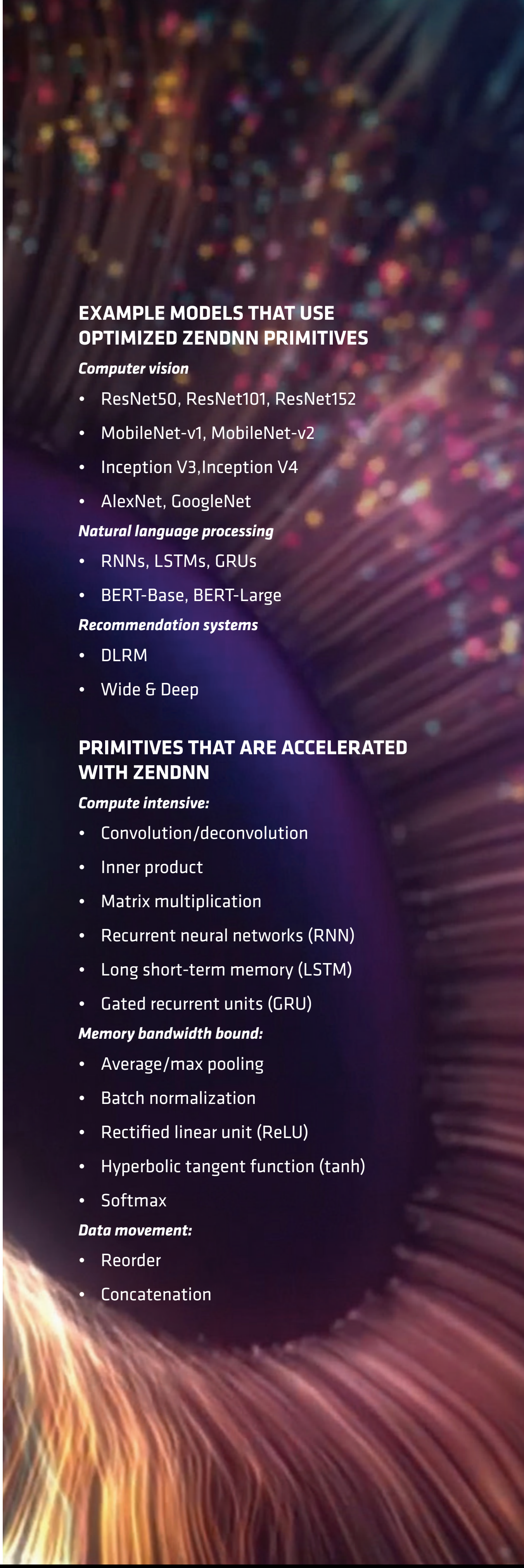
- Machine-learning graph compiler that can handle graphs for all three frameworks
- AMD Optimized CPU Libraries (AOCL), a set of numerical libraries optimized for the 'Zen' core architecture
- ZenDNN, a library of optimized AI primitives
- Zen Software Studio, including AMD Optimizing C/C++ (AOCC) and FORTRAN compilers
- Runtime software for both Microsoft Windows® and Linux

The most important optimization is the ZenDNN library that accelerates deep learning inference workloads on AMD EPYC processors. It resides deep in the software stack and replaces basic AI primitives with equivalent operators that are tuned for AMD EPYC processors (see sidebar). By optimizing at the primitive level, ZenDNN helps to accelerate a broad range of models across diverse application types. It doesn't matter whether you are using a computer vision model to recognize product defects, a natural language application to respond to customer prompts, or an engine to direct proactive maintenance, ZenDNN helps to accelerate a broad range of inferencing workloads.

In addition to optimized primitives, ZenDNN includes graph and framework optimizations that reorganize and restructure models to help them execute more efficiently and better use the compute and memory resources available. As with the optimized primitives, the graph and framework optimizations are not limited to specific use cases, enabling a broader set of models to see gains with ZenDNN.

SOFTWARE PERFORMANCE PROOF

The previous set of performance numbers demonstrated the value of using AMD EPYC processor hardware for AI inferencing. To prove just how much value the UIF (along with the optimized ZenDNN library) brings to all three areas, we benchmarked a few diferent models on the same hardware with and without ZenDNN. The results show dramatic performance increases for all three types of models—computer vision, natural language processing, and recommendation engines. Choose our performance-optimized libraries and see a variety of speedups as shown on the next page.



EXAMPLE MODELS THAT USE OPTIMIZED ZENDNN PRIMITIVES

Computer vision

- ResNet50, ResNet101, ResNet152
- MobileNet-v1, MobileNet-v2
- Inception V3,Inception V4
- AlexNet, GoogleNet

Natural language processing

- RNNs, LSTMs, GRUs
- BERT-Base, BERT-Large

Recommendation systems

- DLRM
- Wide & Deep

PRIMITIVES THAT ARE ACCELERATED WITH ZENDNN

Compute intensive:

- Convolution/deconvolution
- Inner product
- Matrix multiplication
- Recurrent neural networks (RNN)
- Long short-term memory (LSTM)
- Gated recurrent units (GRU)

Memory bandwidth bound:

- Average/max pooling
- Batch normalization
- Rectified linear unit (ReLU)
- Hyperbolic tangent function (tanh)
- Softmax

Data movement:

- Reorder
- Concatenation

ZENDNN OPTIMIZED PRIMITIVES PERFORMANCE PROOF

ACROSS ALL THREE WORKLOADS—COMPUTER VISION, NATURAL LANGUAGE PROCESSING, AND RECOMMENDATION ENGINES—OUR DATA SHOWS THE DRAMATIC SPEEDUP YOU GET WHEN YOU USE THE UNIFIED INFERENCE MODEL WITH TUNED PRIMITIVES IN THE ZENDNN LIBRARY

67%

COMPUTER VISION SPEEDUP[ZD-044]

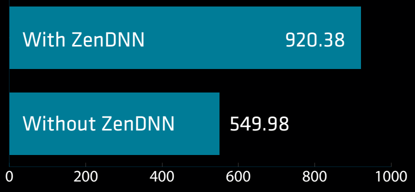
50%

NATURAL LANGUAGE PROCESSING SPEEDUP[ZD-042]

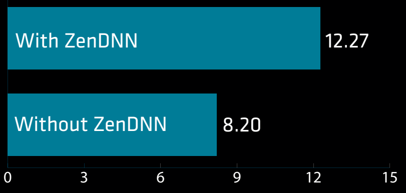
52%

RECOMMENDATION ENGINE SPEEDUP[ZD-043]

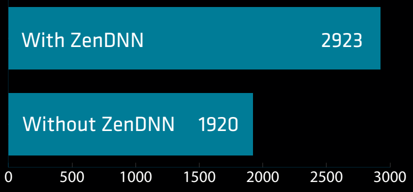
ResNet50 FP32 images/sec, more is better
batch size=640



BERT-Large samples/sec, more is better
batch size=32



DLRM queries/sec, more is better
batch size=1



EACH OF THESE RESULTS COMPARES THE SAME SERVER EQUIPPED WITH 2X 96-CORE AMD EPYC PROCESSORS WITH AND WITHOUT THE ZENDNN LIBRARY

WE HAVE YOU COVERED

Your AI models are trained infrequently. In comparison, inferencing happens every day, minute by minute, across your business. Inferencing needs to be close to your customers and it must deliver the high performance and economics that can help AI transform your business.

Servers powered by AMD EPYC processors provide an excellent platform for CPU-based AI inferencing. With performance propelled by an energy-efficient AVX-512 implementation across 96 cores of processing power, and an optimized library whose primitives drive the processor to deliver its might to your solutions, it's hard to find a better solution. With our Unified Inferencing Frontend, we have you covered. Whether your AI inferencing delivers the performance you need on AMD EPYC processors, servers propelled with AMD Instinct GPU accelerators, or Versal and Zynq adaptive SoCs, we offer the freedom to run your model across our hardware platforms to take advantage of the best that AMD has to offer.

END NOTES

For details on the EPYC footnotes used in this document, visit amd.com/en/claims/epyc4.

EPYC-037 EPYC CPU-powered servers are more often selected for AI Inference Datacenter and Training solutions than Intel Xeon Scalable CPU-based servers posted on MLcommons.org as of 9/8/22. 2nd and 3rd Gen EPYC CPU-based servers power 35 of 63 inference scores in the Closed division and six of 15 Inference Closed-Power division winning ALL of the performance/W records for Image Classification, Object Detection, Medical Imaging, Speech-to-Text, NLP, and Recommendation task workloads Inference list https://mlcommons.org/en/inference-datacenter-21/. EPYC also runs on 73 of 118 platforms shown on the Training list https://mlcommons.org/en/training-normal-20/. The MLPerf™ name and logo are trademarks of MLCommons™ Association ("MLCommons") in the United States and other countries. All rights reserved. Unauthorized use strictly prohibited. See www.mlcommons.org for more information.

EPYC-038 Based on AMD internal testing as of 09/19/2022, geomean performance improvement at the same fixed-frequency on a 4th Gen AMD EPYC™ 9554 CPU compared to a 3rd Gen AMD EPYC™ 7763 CPU using a select set of workloads (33) including est. SPECrate®2017_int_base, est. SPECrate®2017_fp_base, and representative server workloads. SPEC® and SPECrate® are registered trademarks of Standard Performance Evaluation Corporation. Learn more at spec.org.

EPYC-039 The AVX-512 frequency for a given model of the 4th Gen AMD EPYC™ processors are the typical average recorded during AMD internal testing while running an HPC FLOPs workload using AVX-512 binaries in performance determinism mode. AMD's implementation of AVX-512 uses 2-cycle, efficient 256b pipeline and thus could help maintain frequencies higher than AVX2. Actual achievable frequency will vary depending on hardware, software, workloads and other conditions.

EPYC-040 AMD EPYC 9004 CPUs support 12 channels of up to 4800 MHz DDR5 memory which is 460.8 GB/s of maximum memory throughput per socket. Prior generation of EPYC CPUs (7003 series) have a maximum 204.8 GB/s. EPYC 9004 CPUs have 2.25x the memory throughput per CPU. 460.8 ÷ 204.8 = 2.3x (2.25x) the max memory throughput.

ZD-007 Testing conducted by AMD Performance Labs as of Thursday, May 5, 2022, on the ZenDNN v3.3 software library, on test systems comprising of: AMD System: AMD EPYC™ 7763 64-core processors, dual socket, with hyperthreading on, 1500 MHz CPU frequency (max 2450MHz), 256 GB RAM (16x16GB DIMMs @ 2667 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.1 version, kernel version 5.4.0-77-generic x86_64, GCC/G++ version 9.3.0, GNU ID 2.34, Python 3.8.5, AOCC version 3.0, AOCL BLIS version 3.0, TensorFlow version 2.5. Pruning and quantization was performed by the Xilinx Vitis AI pruning and quantization tool v2.0.
For ResNet-50:
- Baseline has 6.9B floating point operations per second (FLOPS)
- Moderately pruned has 4.9B floating point operations per second (FLOPS)
- Aggressively pruned has 2.3B parameters floating point operations per second (FLOPS)

AMD EPYC system running ZenDNN v3.3 library using the Intel public FP32 ResNet-50 (v 2.7) achieves 465 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 baseline FP32 model achieves a ResNet-50 throughput of 523 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 moderately pruned FP32 model achieves a ResNet-50 throughput of 748 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 aggressively pruned FP32 model achieves a ResNet-50 throughput of 1093 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 baseline unpruned and quantized INT8 model achieves a ResNet-50 throughput of 1352 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 moderately pruned and quantized INT8 model achieves a ResNet-50 throughput of 1741 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 aggressively pruned and quantized INT8 model achieves a ResNet-50 throughput of 2664 images/second.

Performance may vary based on use of latest drivers and other factors.

ZD-009 Testing conducted by AMD Performance Labs as of Thursday, May 5, 2022, on the ZenDNN v3.3 software library, on test systems comprising of: AMD System: AMD EPYC™ 7763 64-core processors, dual socket, with hyperthreading on, 1500 MHz CPU frequency (max 2450MHz), 256 GB RAM (16x16GB DIMMs @ 2667 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.1 version, kernel version 5.4.0-77-generic x86_64, GCC/G++ version 9.3.0, GNU ID 2.34, Python 3.8.5, AOCC version 3.0, AOCL BLIS version 3.0, TensorFlow version 2.5. Pruning and quantization was performed by the Xilinx Vitis AI pruning and quantization tool v2.0.
For Inception v3:
- Baseline has 11.43B floating point operations per second (FLOPS)
- Moderately pruned has 9.1B floating point operations per second (FLOPS)
- Aggressively pruned has 6.9B floating point operations per second (FLOPS)

AMD EPYC system running ZenDNN v3.3 library using the Intel public FP32 Inception v3 (v 2.7) achieves 259 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 baseline FP32 model achieves Inception v3 throughput of 309 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 moderately pruned FP32 model achieves Inception v3 throughput of 342 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 aggressively pruned FP32 model achieves Inception v3 throughput of 427 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 baseline unpruned and quantized INT8 model achieves a Inception v3 throughput of 578 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 moderately pruned and quantized INT8 model achieves a Inception v3 throughput of 608 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 aggressively pruned and quantized INT8 model achieves a Inception v3 throughput of 770 images/second.

Performance may vary based on use of latest drivers and other factors.

ZD-010 Testing conducted by AMD Performance Labs as of Thursday, May 5, 2022, on the ZenDNN v3.3 software library, on test systems comprising of: AMD System: AMD EPYC™ 7763 64-core processors, dual socket, with hyperthreading on, 1500 MHz CPU frequency (max 2450MHz), 256 GB RAM (16x16GB DIMMs @ 2667 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.1 version, kernel version 5.4.0-77-generic x86_64, GCC/G++ version 9.3.0, GNU ID 2.34, Python 3.8.5, AOCC version 3.0, AOCL BLIS version 3.0, TensorFlow version 2.5. Pruning and quantization was performed by the Xilinx Vitis AI pruning and quantization tool v2.0.
For VGG16:
- Baseline is 30.9B floating point operations per second (FLOPS)
- Moderately pruned is 17.7B floating point operations per second (FLOPS)
- Aggressively pruned is 15.6B floating point operations per second (FLOPS)

AMD EPYC system running ZenDNN v3.3 library using the baseline FP32 VGG16 model from the Vitis AI Model Zoo achieves 125 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 moderately pruned FP32 model achieves a VGG16 throughput of 216 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 aggressively pruned FP32 model achieves a VGG16 throughput of 230 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 baseline quantized INT8 model achieves a VGG16 throughput of 328 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 moderately pruned and quantized INT8 model achieves a VGG16 throughput of 458 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 aggressively pruned and quantized INT8 model achieves a VGG16 throughput of 475 images/second.

Performance may vary based on use of latest drivers and other factors.

ZD-011 Testing conducted by AMD Performance Labs as of Thursday, May 5, 2022, on the ZenDNN v3.3 software library, on test systems comprising of: AMD System: AMD EPYC™ 7763 64-core processors, dual socket, with hyperthreading on, 1500 MHz CPU frequency (max 2450MHz), 256 GB RAM (16x16GB DIMMs @ 2667 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.1 version, kernel version 5.4.0-77-generic x86_64, GCC/G++ version 9.3.0, GNU ID 2.34, Python 3.8.5, AOCC version 3.0, AOCL BLIS version 3.0, PyTorch version 1.9.0. Pruning was performed by the Xilinx Vitis™ AI pruning and quantization tool v2.0.
For RESNET-50 v1.5:
- Baseline has 8.2B floating point operations per second (FLOPS)
- 30% pruned has 5.8B floating point operations per second (FLOPS)
- 40% pruned has 4.9B floating point operations per second (FLOPS)
- 50% pruned has 4.1B floating point operations per second (FLOPS)
- 60% pruned has 3.3B floating point operations per second (FLOPS)
- -0% pruned has 2.5B floating point operations per second (FLOPS)

AMD EPYC system running ZenDNN v3.3 library using the baseline FP32 RESNET-50 v1.5 model from the Vitis™ AI Model Zoo achieves 363 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 30% pruned FP32 model achieves a RESNET-50 v1.5 throughput of 431 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 40% pruned FP32 model achieves a RESNET-50 v1.5 throughput of 463 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 50% pruned FP32 model achieves a RESNET-50 v1.5 throughput of 481 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 60% pruned FP32 model achieves a RESNET-50 v1.5 throughput of 504 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 70% pruned FP32 model achieves a RESNET-50 v1.5 throughput of 540 images/second.

Performance may vary based on use of latest drivers and other factors.

ZD-012 Testing conducted by AMD Performance Labs as of Thursday, May 5, 2022, on the ZenDNN v3.3 software library, on test systems comprising of: AMD System: AMD EPYC™ 7763 64-core processors, dual socket, with hyperthreading on, 1500 MHz CPU frequency (max 2450MHz), 256 GB RAM (16x16GB DIMMs @ 2667 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.1 version, kernel version 5.4.0-77-generic x86_64, GCC/G++ version 9.3.0, GNU ID 2.34, Python 3.8.5, AOCC version 3.0, AOCL BLIS version 3.0, PyTorch version 1.9.0. Pruning was performed by the Xilinx Vitis™ AI pruning and quantization tool v2.0.
For INCEPTION V3:
- Baseline has 11.42B floating point operations per second (FLOPS)
- 30% pruned has 8.02B floating point operations per second (FLOPS)
- 40% pruned has 6.82B floating point operations per second (FLOPS)

- 50% pruned has 5.68B floating point operations per second (FLOPS)
- 60% pruned has 4.54B floating point operations per second (FLOPS)

AMD EPYC system running ZenDNN v3.3 library using the baseline FP32 INCEPTION V3 model from the Vitis AI Model Zoo achieves 275 images/second.

- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 30% pruned FP32 model achieves a INCEPTION V3 throughput of 247 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 40% pruned FP32 model achieves a INCEPTION V3 throughput of 284 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 50% pruned FP32 model achieves a INCEPTION V3 throughput of 330 images/second.
- The same AMD EPYC system with the Xilinx Vitis™ AI Model Zoo 2.0 60% pruned FP32 model achieves a INCEPTION V3 throughput of 389 images/second.

Performance may vary based on use of latest drivers and other factors.

ZD-031 Testing conducted by AMD Performance Labs as of Tuesday, December 13, 2022, on the ZenDNN v4.0 software library, on test systems comprising of: AMD EPYC™ 9654 96-core processor, dual socket, with hyperthreading on, 2400 MHz CPU frequency (Max 3700 MHz), 1.5TB RAM (24 x 64GB DIMMs @ 4800 MT/s; DDR5 synchronous registered), NPS4 mode, Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-132-generic, BIOS RTI1001C compared to: AMD EPYC 7763 64-core processors, dual socket, with hyperthreading on, 2450 MHz CPU frequency (max 3500 MHz), 512 GB RAM (32 x 64GB DIMMS @ 2900 MT/s; DDR4 synchronous registered), Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-135-generic, BIOS RYM1008B. Both systems utilized: NPS4 mode, GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0. Performance may vary based on use of latest drivers and other factors.

ZD-036 Testing conducted by AMD Performance Labs as of Thursday, January 12, 2023, on the ZenDNN v4.0 software library, Xilinx Vitis AI Model Zoo 3.0, on test systems comprising of AMD Eng Sample of the EPYC 9004 96-core processor, dual socket, with hyperthreading on, 2150 MHz CPU frequency (Max 3700 MHz), 786GB RAM (12 x 64GB DIMMs @ 4800 MT/s; DDR5 - 4800MHz 288-pin Low Profile ECC Registered RDIMM 2RX4), NPS1 mode, Ubuntu® 20.04.5 LTS version, kernel version 5.4.0-131-generic, BIOS TQZ1000F, GCC/G++ version 11.1.0, GNU ID 2.31, Python 3.8.15, AOCC version 4.0, AOCL BLIS version 4.0, TensorFlow version 2.10. Pruning was performed by the Xilinx Vitis AI pruning and quantization tool v3.0. Performance may vary based on use of latest drivers and other factors.

ZD-037 Testing conducted by AMD Performance Labs as of Tuesday, December 13, 2022, on the ZenDNN v4.0 software library, on test systems comprising of AMD EPYC™ 9654 96-core processor, dual socket, with hyperthreading on, 2400 MHz CPU frequency (Max 3700 MHz), 1.5TB RAM (24 x 64GB DIMMs @ 4800 MT/s; DDR5 synchronous registered), NPS4 mode, Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-132-generic, BIOS RTI1001C , GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0, compared to AMD EPYC 7763 64-core processors, dual socket, with hyperthreading on, 2450 MHz CPU frequency (max 3500 MHz), 512 GB RAM (32 x 64GB DIMMS @ 2900 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-135-generic, BIOS RYM1008B, GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0. For ResNet-50, with a batch size of 640. Performance may vary based on use of latest drivers and other factors.

ZD-038 Testing conducted by AMD Performance Labs as of Tuesday, December 13, 2022, on the ZenDNN v4.0 software library, on test systems comprising of AMD EPYC™ 9654 96-core processor, dual socket, with hyperthreading on, 2400 MHz CPU frequency (Max 3700 MHz), 1.5TB RAM (24 x 64GB DIMMs @ 4800 MT/s; DDR5 synchronous registered), NPS4 mode, Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-132-generic, BIOS RTI1001C , GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0, compared to AMD EPYC 7763 64-core processors, dual socket, with hyperthreading on, 2450 MHz CPU frequency (max 3500 MHz), 512 GB RAM (32 x 64GB DIMMS @ 2900 MT/s; DDR4 synchronous registered), NPS4 mode, Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-135-generic, BIOS RYM1008B, GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0. Performance may vary based on use of latest drivers and other factors.

ZD-042 Testing conducted by AMD Performance Labs as of Tuesday, December 13, 2022, on the ZenDNN v4.0 software library, using BERT-Large (FP32), with a batch size of 32 and sequence length of 512, on test systems comprising of: AMD System: AMD EPYC™ 9654 96-core processor, dual socket, with hyperthreading on, 2400 MHz CPU frequency (Max 3700 MHz), 1.5TB RAM (24 x 64GB DIMMs @ 4800 MT/s; DDR5 synchronous registered), NPS4 mode, Ubuntu® 200.04.4 LTS version, kernel version 5.4.0-132-generic, BIOS RTI1001C , GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0. Results may vary based on factors such as software versions and BIOS settings.

ZD-043 Testing conducted by AMD Performance Labs as of Tuesday, February 7, 2023, on the ZenDNN v4.0 software library, running FP32 DLRM with a batch size of 1 on test systems comprising of: AMD System: AMD Eng Sample of the AMD EPYC™ 9004 96-core processor, dual socket, with hyperthreading on, 2150 MHz CPU frequency (Max 3700 MHz), 786GB RAM, 768 L3 Cache, NPS1 mode, Ubuntu® 20.04.5 LTS version, kernel version 5.4.0-131-generic, BIOS TQZ1000F, GCC/G++ version 11.1.0, GNU ID 2.31, Python 3.8.15, AOCC version 4.0, AOCL BLIS version 4.0, TensorFlow version 2.10. Results may vary based on factors such as software versions and BIOS settings. ZD-043.

ZD-044 Testing conducted by AMD Performance Labs as of Tuesday, December 13, 2022, on the ZenDNN v4.0 software library using FP32 ResNet50, batch size 640, on test systems comprising of: AMD System: AMD EPYC™ 9654 96-core processor, dual socket, with hyperthreading on, 2400 MHz CPU frequency (Max 3700 MHz), 1.5TB RAM (24 x 64GB DIMMs @ 4800 MT/s; DDR5 synchronous registered), NPS4 mode, Ubuntu® 20.04.4 LTS version, kernel version 5.4.0-132-generic, BIOS RTI1001C , GCC/G++ version 9.4.0, GNU ID 2.34, Python 3.8, AOCL BLIS version 4.0, TensorFlow version 2.10, PyTorch version 1.12.0. Results may vary according to factors including software versions and BIOS settings.