Intel® Xeon Phi™ Delivers Competitive Performance For Deep Learning—And Getting Better Fast

By Andres R. (Intel), Added September 26, 2016

Authors: Dheevatsa Mudigere, Dipankar Das, Vadim Pirogov, Murat Guney, Srinivas Sridharan, and Andres Rodriguez of Intel Corporation

Baidu's recently announced deep learning benchmark, DeepBench, documents performance for the lowest-level compute and communication primitives for deep learning (DL) applications. The goal is to provide a standard benchmark to evaluate different hardware platforms using the vendor's DL libraries.

Intel continues to optimize its Intel® Xeon and Intel® Xeon Phi™ processors for DL via the Intel Math Kernel Library (Intel MKL). Intel MKL 2017 includes a collection of performance primitives for DL applications. The library supports the most commonly used primitives necessary to accelerate image recognition topologies and GEMM primitives necessary to accelerate various types of RNNs. The functionality includes convolution, inner product, pooling, normalization and activation primitives, with support for forward (inference) and backward (gradient propagation) operations. The MKL 2017 library is freely available with a community license, and some of these optimizations are also available as part of open source Intel MKL-DNN project.

Intel® Xeon Phi™ processors were used for this benchmark. In this paper, we point out the DL operations where Intel® Xeon Phi™ shines—and is rapidly improving.

DeepBench background

DeepBench aims to include primitives such GEMMs (General Matrix-Matrix Multiplication), convolution layers, and recurrent layers with specific configurations used across different type of networks/applications. The current release is a first attempt at this and is not yet a complete set—the hope is that with active participation from the community this will become a comprehensive benchmark of primitives used in deep neural networks (DNNs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory networks (LSTMs) across a number of applications such as image and speech recognition, and natural language processing (NLP).

Further, DeepBench includes cases with varying mini-batch sizes to capture the impact of scaling (data parallelism). DeepBench is primarily intended as a tool for comparing different hardware platforms. The metrics Baidu published is absolute performance (in terms of TeraFLOPS/s).

In the remainder of this paper, we address the performance and suitability of Intel Xeon Phi processors for various DL operations.

GEMM

The majority of the computations involved in deep learning can be formulated as matrix-matrix multiplication operations, making GEMM a core compute primitive. Typically, the matrices coming from DL applications – fully connected (FC) layers, recurrent neural networks (RNNs), and long short-term memory networks (LSTMs)—are skewed and small, resulting in dense, small and oddly shaped (tall-skinny, fat-short) GEMM operations. This is captured by the GEMM kernel test cases in DeepBench. Intel Math Kernel Library (Intel MKL) 2017 includes optimized GEMM implementations that achieve high performance on both the Intel Xeon processor and Intel Xeon Phi processor for matrices that are typically seen in DL applications, exposed through the packed GEMM application programming interface (API).

Unlike the conventional GEMM operations (large and square matrices), these dense, small, and oddly shaped operations are particularly challenging due to their limited parallelism and highly skewed dimensions. Conventional methods fail to achieve peak performance, as outlined by this Baidu paper on optimizing GEMM performance for RNNs [1].

The specialized packed API implements an optimized block-GEMM operation, with a block formulation to increase reuse of blocks, without additional data rearrangement and fine-grained parallelization with minimal on-demand synchronization to increase the available concurrency for small matrices. These optimizations allow designers to effectively exploit the full-cache hierarchy in both Intel® Xeon and Intel® Xeon Phi™, extracting sufficient parallelism to ensure that all cores are kept busy to achieve significantly improved (near-peak) performance for such typical DL matrices.

The DeepBench GEMM kernel results on the Intel Xeon Phi processor include both the conventional Intel MKL GEMM and also the new packed GEMM API. These numbers are measured on Intel Xeon Phi processor 7250 (codenamed Knights Landing, or KNL) with Intel MKL 2017, which is publicly available. It can be seen from the DeepBench GEMM results (Fig. 1) (Nvidia performance measured by Baidu) that Intel Xeon Phi processor performance is higher than the performance of the Nvidia* M40 GPU (whose peak FLOPs are comparable with Intel Xeon Phi processor) across almost every single configuration, and is higher than the performance on the Nvidia* Pascal TitanX across some smaller, medium (N <= 64) matrices. With the next generation of Intel Xeon Phi processor (codenamed Knights Mill) offering significantly higher raw compute power, we would expect to see better performance when it is released next year.

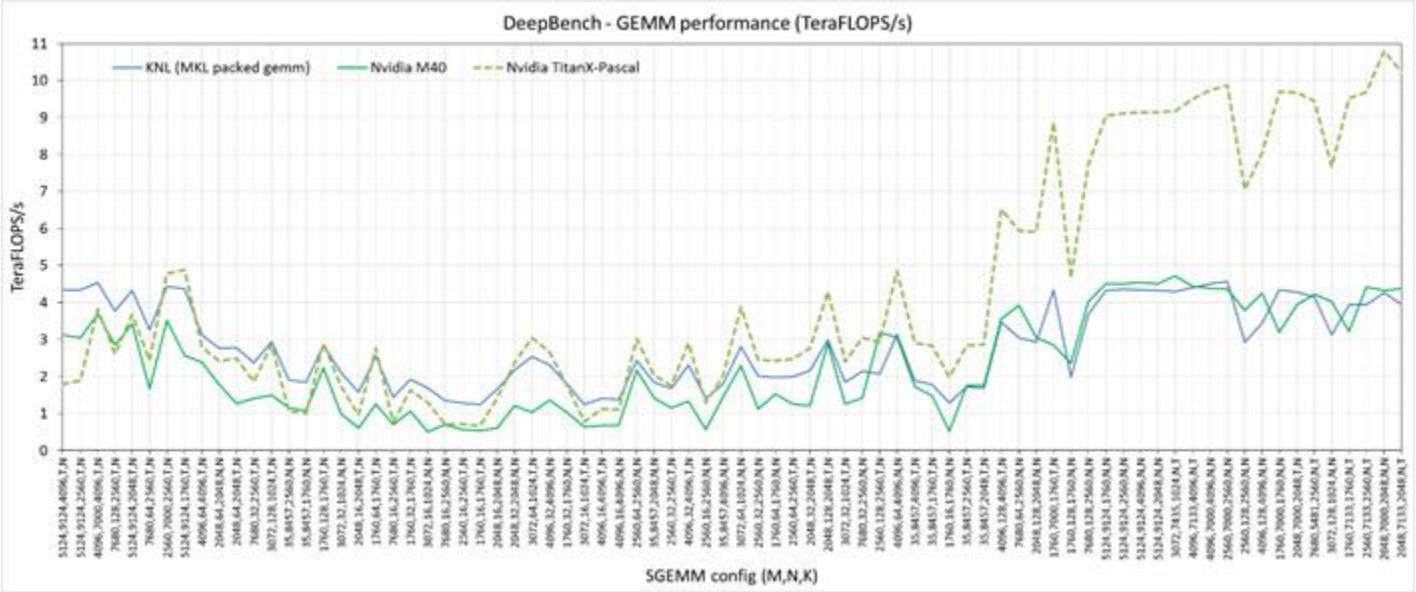
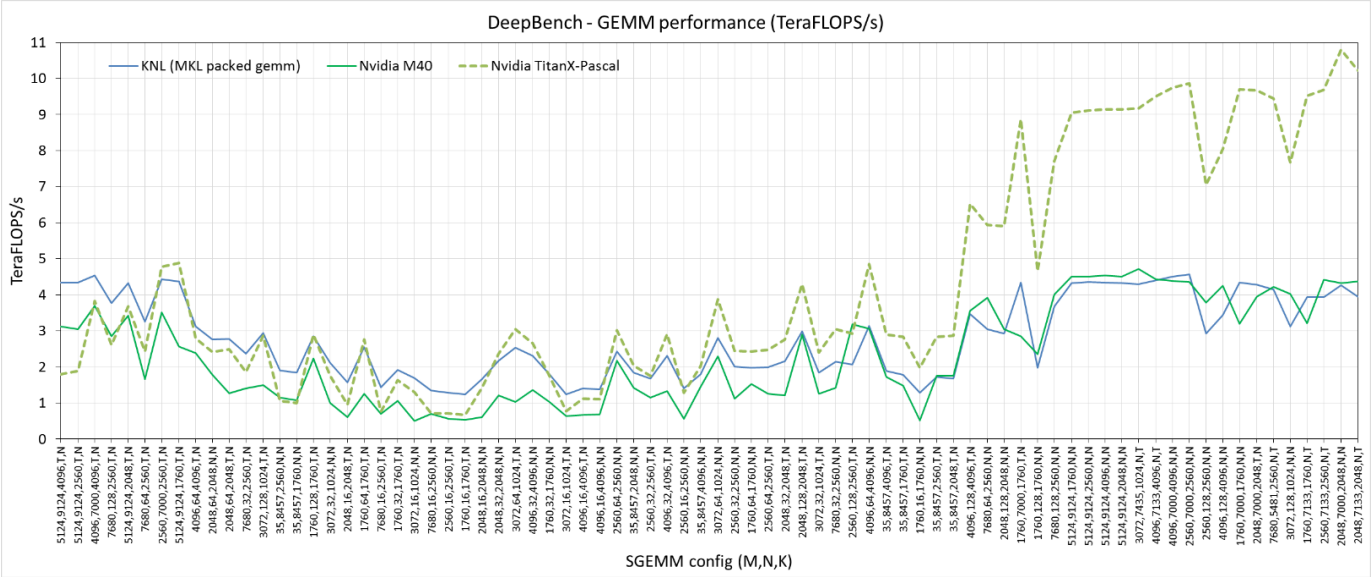


Fig. 1 Source: Data from Baidu as of Sept 26, 2016

View Larger Image
Close



Convolution

The convolution operation is the other primary compute kernel in DL applications. For image-based application (CNNs), convolution layers contribute to the majority of the compute task. Also, increasingly, convolutional layers are also being used for speech and NLP (as acoustic models) applications, as well.

The convolutional-layer operation consists of a six-level nested loop over output feature maps (K), input feature maps (C), height and width of feature map (H, W), and kernel height and widths (R, S). Additionally, this operation is done over all the samples of the mini-batch (N). Hence, for typical layer configurations, this can result in significant computation.

This operation written as nested *for* loops in naïve order, does not leverage the available reuse (weights remaining unchanged across recurrent iterations) and becomes limited by memory bandwidth, thus not utilizing all the available compute on the Intel Xeon Phi processor. MKL offers optimized implementation of the convolution layers using the direct convolution operator, which gets close to achievable peak performance.

The direct convolution kernel includes the following optimizations:

- Reformulating the convolution operation to better utilize the cache hierarchy. The loop-over output and input feature maps (K, C) are blocked for on-die caches and allow for inner-loop vectorization over output feature maps and independent fused multiply-add computations
- Data is laid out so that the innermost loop accesses are contiguous, ensuring better utilization of cache lines and, therefore, bandwidth, while also improving prefetcher performance
- Register blocking to improve reuse of data in register files, decrease on-core cache traffic, and hide the latency of the FMA operations
- Optimal work partitioning to ensure that all cores are busy, with minimal load imbalance.

More detailed information on the implementation detail can be found under the machine learning chapter of the Intel Xeon Phi processor reference book [2].

These numbers are measured on Intel Xeon Phi processor 7250 with Intel MKL 2017. For the DeepBench convolution kernel (Fig. 2) we also include results for the open-source Intel optimized convolution layer implementation using libxsmm [3]. The absolute performance for the convolution layers are competitive compared to the Nvidia M40 (with comparative FLOPs). However, since the current version of Intel MKL supports only the direct convolution operator, the marked kernels with larger differences are for those convolutions layers where the direct convolution kernels on Intel Xeon Phi processor are being compared to the Winograd-based implementation. Intel MKL does not provide optimized Winograd convolution implementations at this point. Incorporating the Winograd-based implementation into MKL is work-in-progress. Despite the fact that the Winograd convolution algorithm shows significant speedups on certain convolution shapes and sizes, it does not significantly contribute to full topology performance.

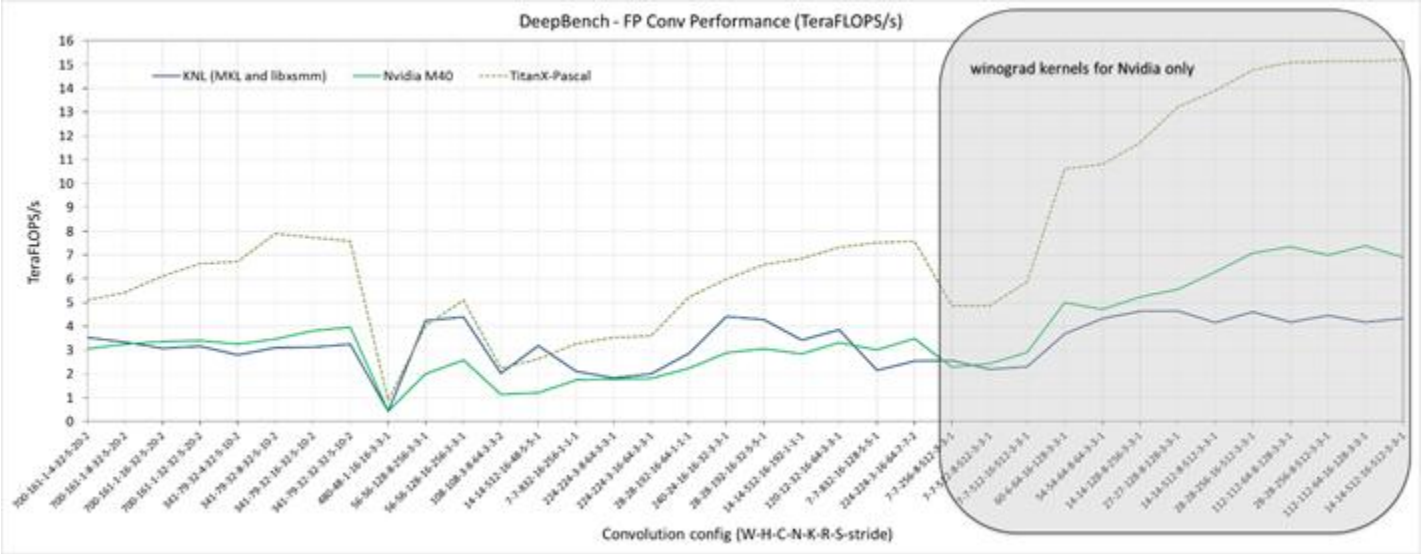
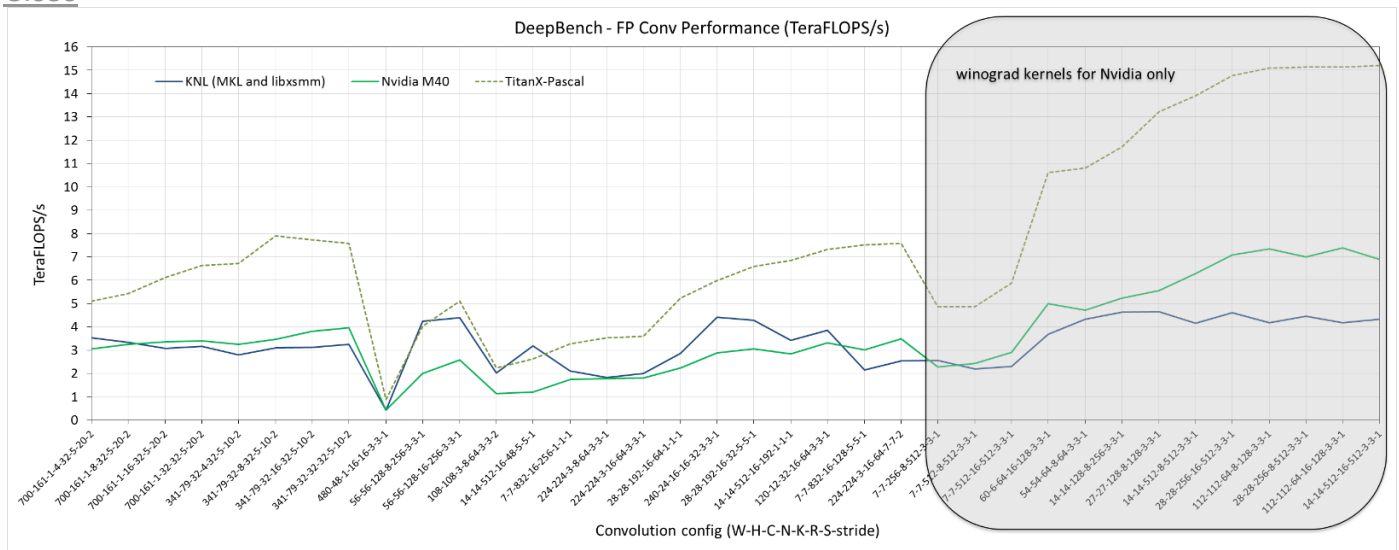


Fig. 2 Source: Data from Baidu as of Sept 26, 2016

View Larger Image
Close



AllReduce

AllReduce is the communications primitive in DeepBench that covers message sizes commonly seen in deep learning networks and applications. This benchmark consists of measuring MPI_AllReduce latencies for five different message sizes (in floats): 100K, 3MB, 4MB, 6.4MB, and 16MB on 2, 4, 8, 16, and 32 nodes. This uses the AllReduce benchmark from the Ohio State University micro-benchmarks suite [4], with minor modifications.

We report the MPI_AllReduce time measured on Intel Xeon Phi processor 7250 on our internal Endeavor cluster with Intel® Omni-Path Architecture (Intel® OPA) series 100 fabric with fat-

tree topology, using Intel MPI 5.1.3.181. The competitive data from Baidu is measured on a GPU cluster with 8 NVIDIA TitanX-Maxwell cards per node (with optimized intra-node comms). This is compared with a single Intel Xeon Phi processor consisting of a node, so a 32-node Intel Xeon Phi processor measurement below is comparable to 4 GPGPU nodes, with each node having a maximum of 8 cards.

The Intel Xeon Phi processor DeepBench results are with the stock Intel MPI Library 2017 on the above-mentioned cluster (Fig. 3). The latencies are better for 8 GPUs (in a single node) compared to 8 Intel Xeon Phi processor-based nodes since this constitutes to within node (peer-to-peer) communication. However, for communication across nodes, the Intel Xeon Phi processor AllReduce latencies are significantly better. Latencies for 16 Intel Xeon Phi processor-based nodes were better than that for 2 GPU (x8) nodes across most message sizes. Fig. 3 is normalized against TitanX-Maxwell, which Baidu measured as being number-1.

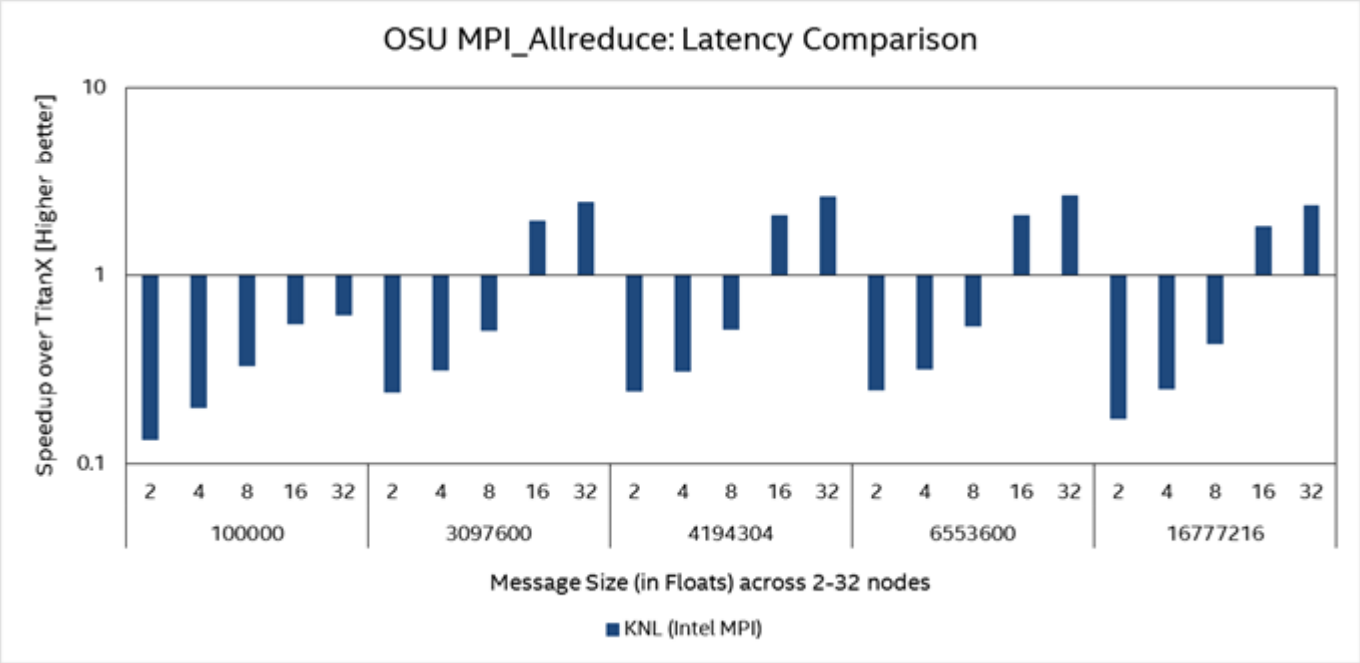
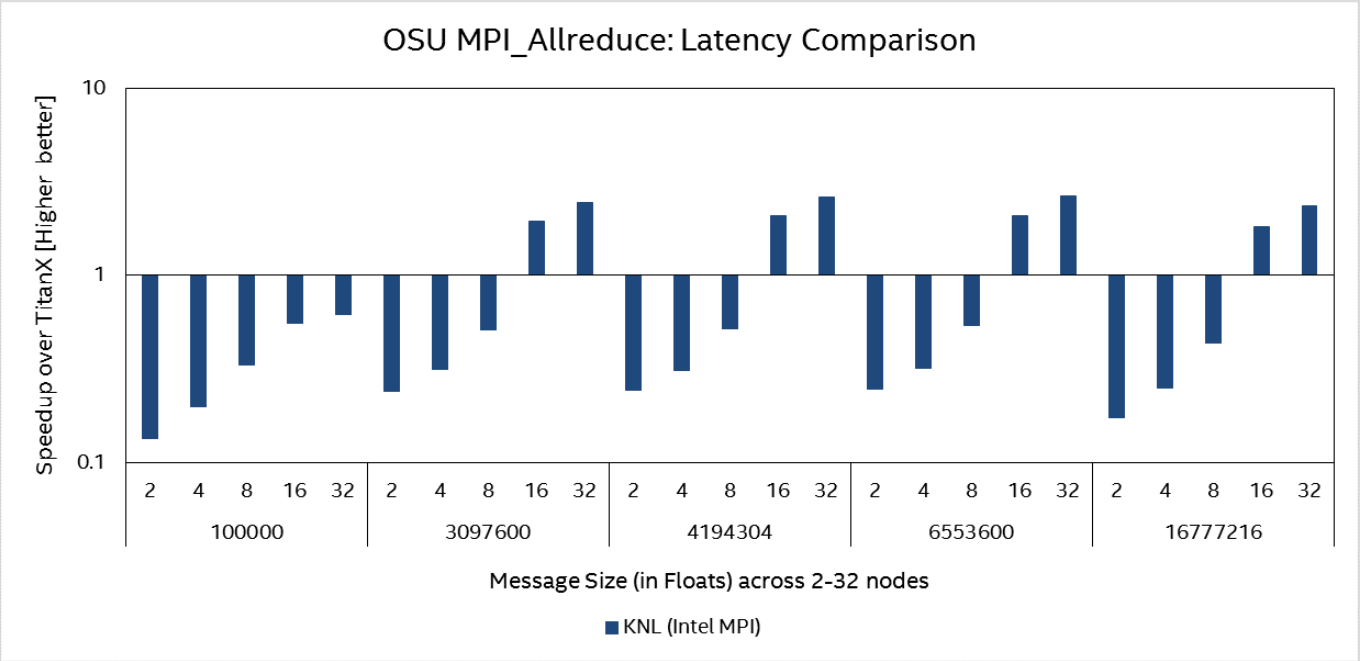


Fig. 3 Source: Data from Baidu as of Sept 26, 2016

View Larger Image
Close



Further, we also present results using our optimized communication library (which was also presented in [Pradeep Dubey's IDF16 Technical Session)] which further improves AllReduce latencies by an average of 3.5X across the message sizes and node counts of interest (Fig. 4). This benchmark only captures the latency for a single MPI_AllReduce operation. Typically, in any application context we can expect to have multiple such operations in flight, and in such situations we can expect to see further performance improvements.

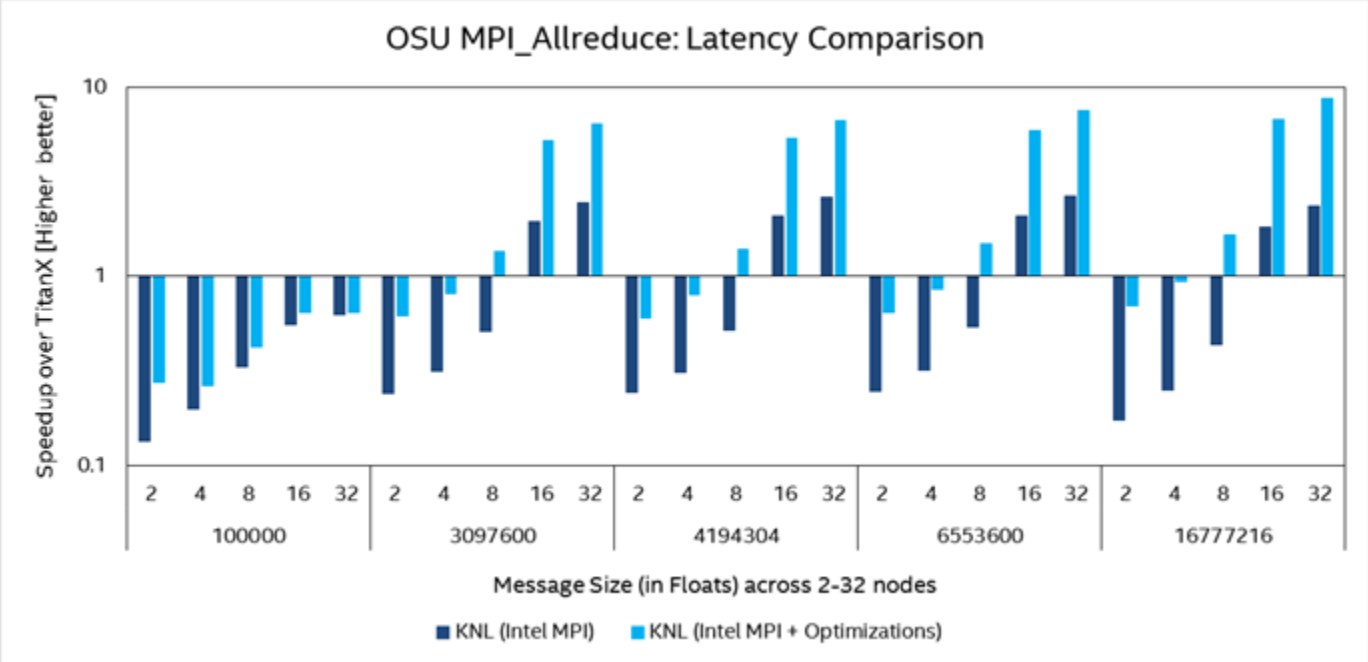
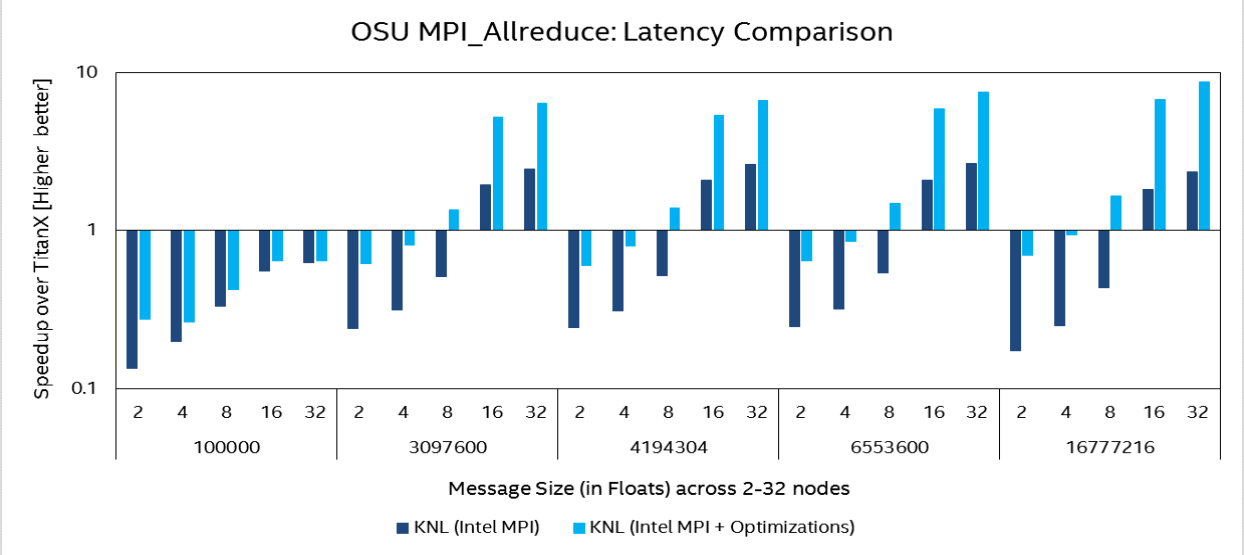


Fig. 4 Source: Intel internal measurements, September 2016 [5]

View Larger Image
Close



Recurrent Layers – RNN/LSTM

DeepBench also includes recurrent layers—vanilla RNN and LSTM layers, primarily based on the DeepSpeech2 model configurations and for different mini-batch sizes—to capture the impact of scaling. The core compute kernel for these recurrent layers are still the GEMM operations, and the matrix sizes corresponding to these layers are already captured in the GEMM benchmark. For these cases, we can see from Fig. 1 that Intel Xeon Phi consistently performs better than current Nvida Maxwell GPUs and in many cases also better than Nvidia Pascal GPU which has almost 2X more peak flops. However, these layers are included as independent primitives to showcase RNN/LSTM-specific optimizations. For the current benchmark release, we do not include Intel Xeon Phi processor results for these cases. We are working on an optimized implementation for the RNN layers, which exploits the specific usage pattern to more efficiently leverage available caches. The Intel Xeon Phi processor results for these cases will be added once RNN layers support is introduced to Intel MKL.

While these benchmark results are a snapshot in time, Intel continues to invest in software optimizations that would further improve the performance of Intel Xeon and Intel Xeon Phi family processors especially on the convolution benchmark. Intel will continue to update the results to ensure end customers have a choice of silicon when it comes to deep learning workloads.

Authors

Dheevatsa Mudigere, Dipankar Das, Vadim Pirogov, Murat Guney, Srinivas Sridharan, and Andres Rodriguez, Intel Corporation

[1] http://svail.github.io/rnn_perf/

[2] http://lotsofcores.com/KNLbook

[3] https://github.com/hfp/libxsmm

[4] http://mvapich.cse.ohio-state.edu/benchmarks/

[5] FTC Disclaimer: Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit: **www.intel.com/benchmarks**.

Configuration: Intel® Xeon Phi™ Processor 7250 (68 Cores, 1.4 GHz, 16GB MCDRAM), 128 GB DDR4-2400 MHz, Intel® Omni-Path Host Fabric Interface Adapter 100 Series 1 Port, Red Hat Enterprise Linux 6.7, Intel® ICC version 16.0.3, Intel® MPI Library 5.1.3 for Linux, Intel® Optimized DNN Framework*

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804

Source:

https://software.intel.com/en-us/articles/intel-xeon-phi-delivers-competitive-performance-for-deep-learning-and-getting-better-fast