PetaVision Neural Simulation Toolbox on Intel KNLs

BORAM YOON, Los Alamos National Laboratory, USA
PETE SCHULTZ, Los Alamos National Laboratory, USA and New Mexico Consortium, USA
GARRETT KENYON, Los Alamos National Laboratory, USA and New Mexico Consortium, USA

PetaVision, an open source neural simulation toolbox, is optimized for Intel Knights Landing processors on Trinity supercomputer. The optimization is performed both for the single-node and multi-node performance, and the code scaled up to 8192 KNL nodes, efficiently. Using the optimized PetaVision, we implemented a large-scale neuromorphic algorithm called the Sparse Prediction Machine (SPM), on the Los Alamos Trinity supercomputer, and trained the SPM to predict 8th frame from the preceding 7 frames on ImageNet video.

CCS Concepts: • **Computing methodologies** → **Machine learning**; • **Hardware** → **Emerging architectures**;

Additional Key Words and Phrases: Sparse coding, unsupervised learning, intel knights landing, trinity supercomputer

1 EXTENDED ABSTRACT

PetaVision is an open source neural simulation tool box [1], specialized on the sparse coding unsupervised learning [2]. The program is written in C++, and provides parallelization via MPI and OpenMP or CUDA. We optimized performance of the PetaVision for Intel Knights Landing (KNL) architecture. KNL is the code name of Intel's second-generation Xeon Phi processors. KNL enables massive parallelism via many-core architecture up to 72 cores per socket with four threads per core, wide vector operations with AVX-512 instructions, and efficient memory access through the high-bandwidth MCDRAM. Many supercomputing facilities are adopting the KNL systems. In this study, we specifically targeted the Trinity cluster at Los Alamos National Laboratory (LANL), which is a Cray XC40 system with about 9500 KNL nodes and about 9500 Haswell nodes.

The optimization is performed in two steps: (1)single-node optimization and (2)multi-node optimization. For the single-node optimization, we extracted the bottleneck calculation, computing convolution of two vectors corresponds to the neuron activities and weights, as a stand-alone program. On the kernel, we applied various optimization techniques for KNL processors, such as vectorization, loop reordering and tiling, and memory aligning. The performance improvements are shown in Figure 1. We also collapsed for-loops to improve threading efficiency, and we optimized OpenMP scheduling to reduce load imbalance and OpenMP overhead.

After single-node optimization, we improved multi-node performance. PetaVision has two-level parallelization. The input images are gridded, and MPI tasks are spread over the grid so that each MPI rank works on a cell. Most of the MPI communications occurs between immediate neighbors in the 2D grid. Additional parallelization is obtained by processing multiple batches of images simultaneously. After processing each image, there is a global MPI communication that synchronizes neurons across the batches. For multi-node optimization, we made the MPI communications as non-blocking as possible. We also overlapped calculations with MPI communications, and implemented parallel I/O. The scaling behavior of the PetaVision is shown in Figure 2.

Using the optimized PetaVision, we implemented a large-scale neuromorphic algorithm called the Sparse Prediction Machine (SPM). The goal of an SPM is to predict future states of a system from a sequence of previous states, or in the case of video, to predict a subsequent frame from previous frames. Detailed description of the SPM is given in Figure 3.

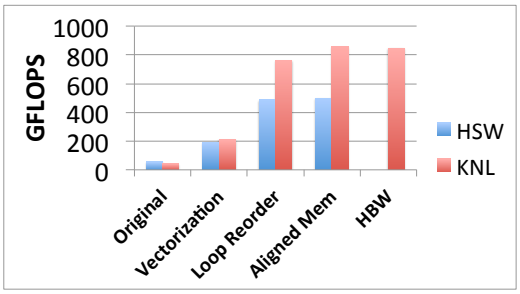

Fig. 1. Performance improvements for each optimization technique applied to the hotspot kernel. The test is performed on a KNL or Haswell node, only with OpenMP. Since the Haswell node has two sockets, the performance is underestimated, and will perform better if the job is distributed in two NUMA nodes. The performance varies with run parameters such as number of neurons.

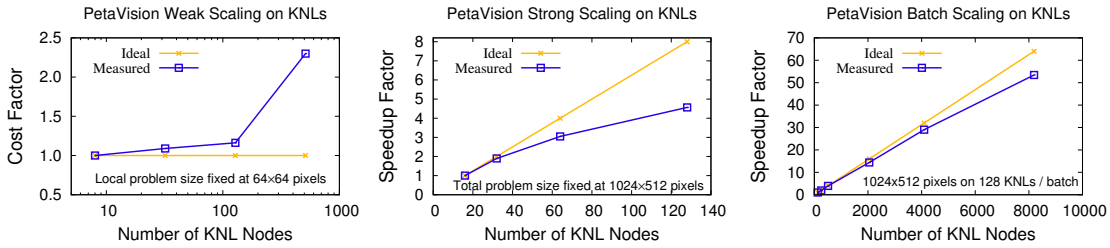

Fig. 2. PetaVision scaling plots on the Trinity cluster. Cost factor is the wall-clock time compared to 8-node run, and the Speedup factor is the inverse wall-clock time compared to the smallest (16 nodes on strong, and 128 nodes on batch scalings) run.

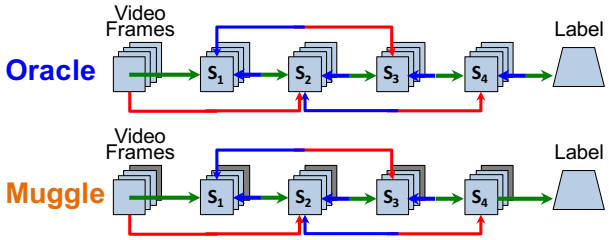

Fig. 3. Sparse Prediction Machine (SPM) consist of a pair of deep sparse generative autoencoders (DSGAs), denoted Muggle and Oracle. Only the Oracle can see frames that are in the future (shaded gray in Muggle) and only the Oracle can see the Label, if present. The Muggle and Oracle otherwise have identical architectures and share all connections. The cost function promotes the learning of features that encourage the Muggle to generate sparse representations that match those of the Oracle.

We trained a SPM using ImageNet video frames on the Trinity cluster. The trained SPM was able to predict the 8th frame from the preceding 7 frames, including successfully separating foreground and background motion.

REFERENCES

[1] PetaVision. 2017. https://github.com/PetaVision. (2017).

[2] Christopher J. Rozell, Don H. Johnson, Richard G. Baraniuk, and Bruno A. Olshausen. 2008. Sparse Coding via Thresholding and Local Competition in Neural Circuits. *Neural Comput.* 20, 10 (Oct. 2008), 2526–2563. https://doi.org/10.1162/neco.2008.03-07-486