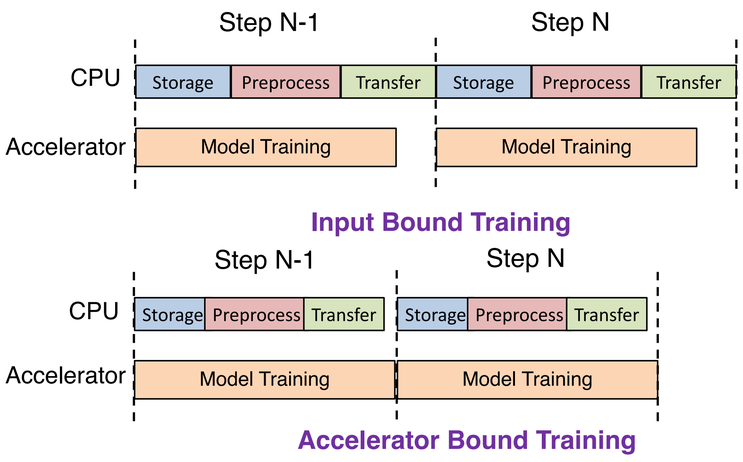
What Can Intelligent SSDs Do for Machine Learning?

Jinyoung Choi, Yu-Ching Hu, Abenezer Wudenhe and Hung-Wei Tseng
Extreme Storage & Computer Architecture Laboratory, Department of Computer Science | North Carolina State University

Objective

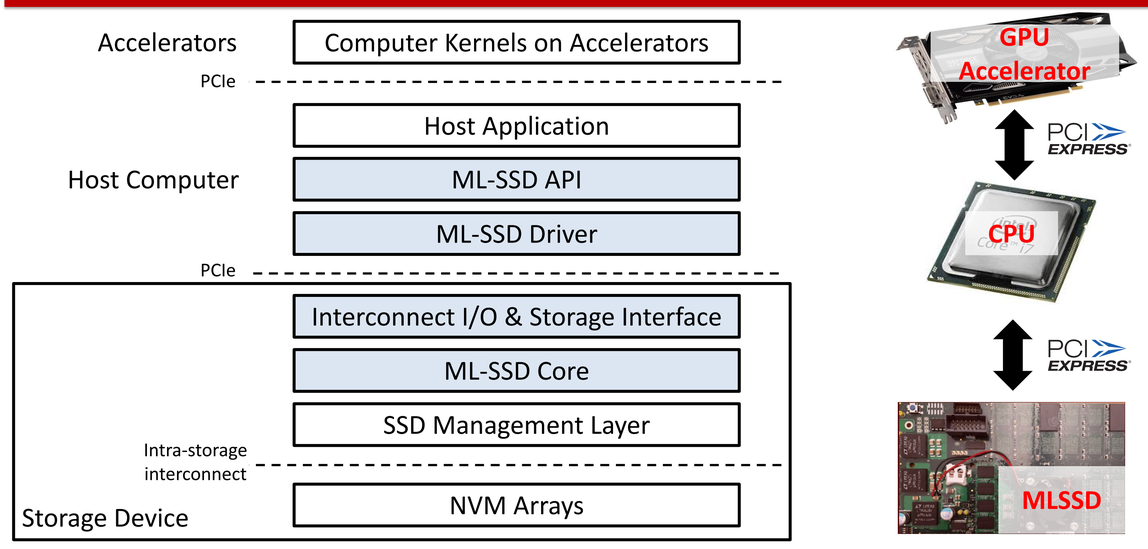


Increase in accelerator performance for Machine Learning (ML) applications means data input becomes the critical path for ML training. Usually, data input consists of retrieving data from storage to the CPU, preprocess operations such as shuffle, and finally transferring data to the ML accelerator. Resulting in:

- Underutilizing Accelerators
- Longer Training Time

This project investigates intelligent storage devices for addressing the input/shuffling preprocess overhead by proposing ML-SSD.

ML-SSD: The System Architecture and Design



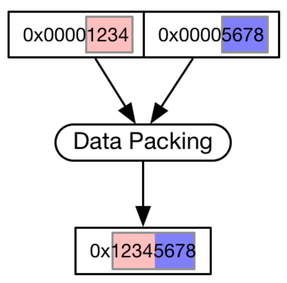
MLSSD in a heterogeneous computer system

The core MLSSD layer resides inside the storage device to change data resolutions presented to applications. MLSSD interacts with existing system I/O interfaces to sup-port the extended interface for resolution adjustments. MLSSD also works together with the SSD management layer to locate the desired data. The host system needs an extended kernel driver, and API functions, for the applications to send requests and exchange data. MLSSD implements the data shuffling feature in the kernel driver and API layer.

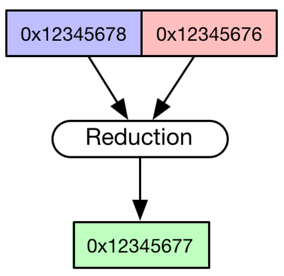
Core MLSSD Layer

The current MLSSD framework supports the following 3 categories of operators working on various data types to adjust the resolution to achieve the best performance.

Data Packing The data packing operator trims the dataset size by using fewer bytes to express each item and condenses the layout in memory. This operator is suitable for datasets using only a small range within the number space of the original data type, or the applications which can tolerate some inaccuracy in the input data, e.g. FP64 to FP32.



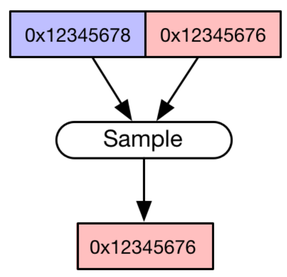
Data Packing Operator



Reduction Operator

Reduction The reduction operator applies a function (e.g., average) to groups of input values, usually neighboring data items in the raw data, and generate a single output for each group. Thus, MLSSD only sends out the resulting values of each group to reduce the amount of data going through the system interconnect.

Sample The sample operator chooses a subset of items from raw data and sends the selected items to the host computer. This operators can perform uniform data selection, random data selection, or only report the most representative data. The sample operator helps filter repetitive or similar inputs. If the compute kernel is elastic to the number of records within the dataset, the sample operator can achieve the same effect as loop perforation without any code modification.



Sampling Operator

The prototype SSD

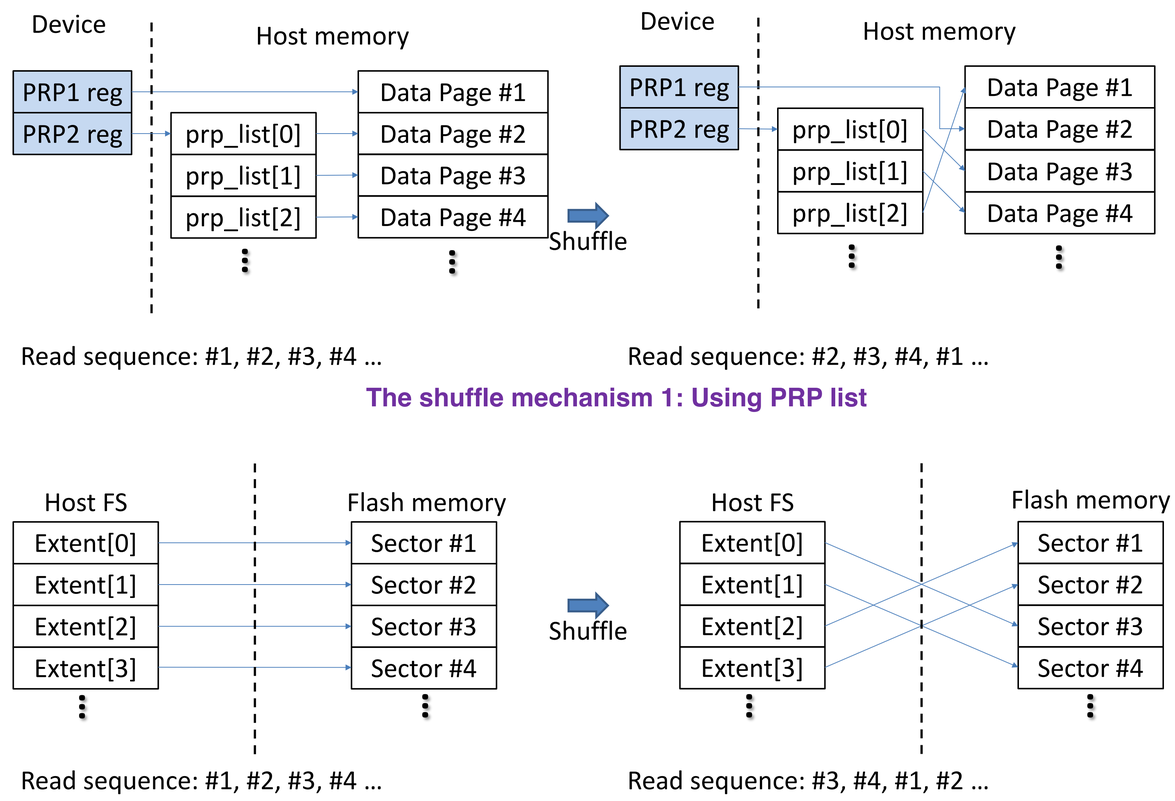
The prototype SSD with the proposed ML-assisted layer

We build a MLSSD by extending a commercialized datacenter-class SSD. We extended the NVMe driver in this system to support additional MLSSD NVMe commands. This SSD runs our modified firmware programs which is also compatible with standard NVMe. Throughout our tests, the baseline SSD achieves 3.2 GB/s bandwidth to the host system.

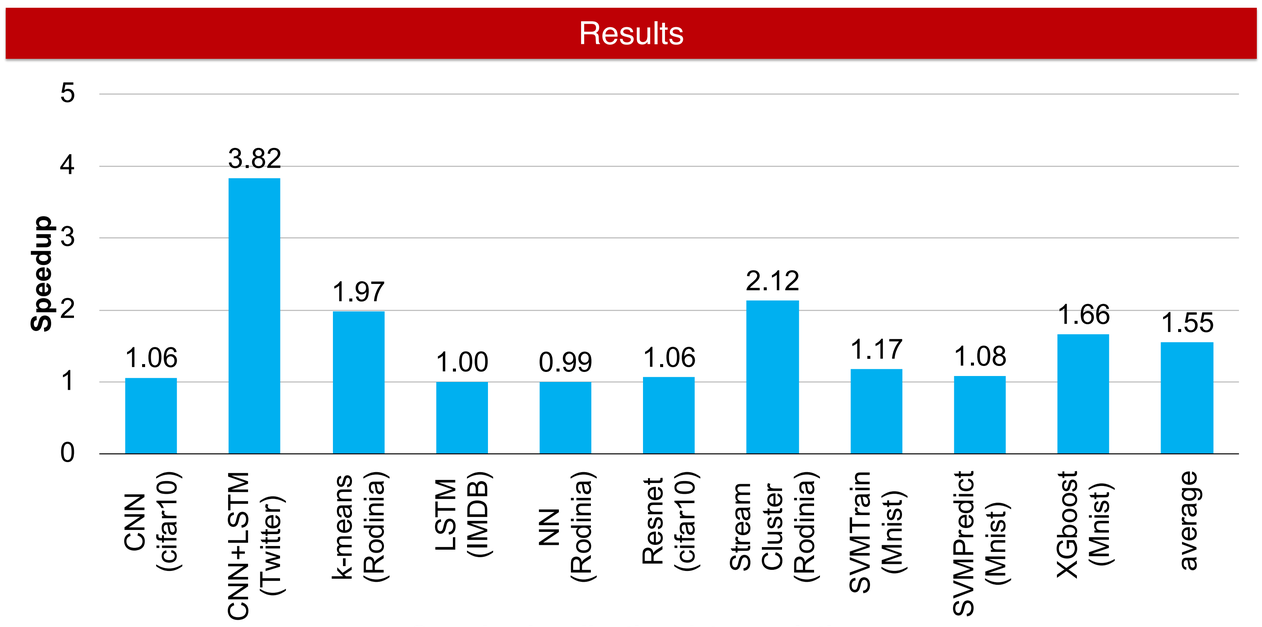
Extended NVMe Stack for Machine Learning

mlssd_read(HostMem, size)	This API shuffles data pages and stores it in the 'HostMem' buffer for the 'size'. We can avoid redundant data copies by utilizing NVMe's PRP list data structure and extents in Ext4 file system.
mlssd_get_sector(fd)	This API produces sector lists of the file('fd') based on the logical block address(LBA) and the number of blocks within the extents.

ML-SSD API

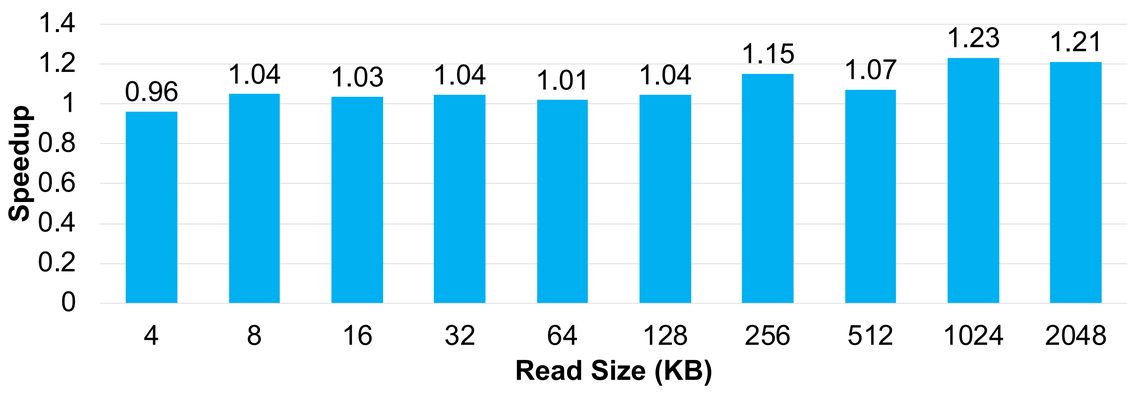


Results



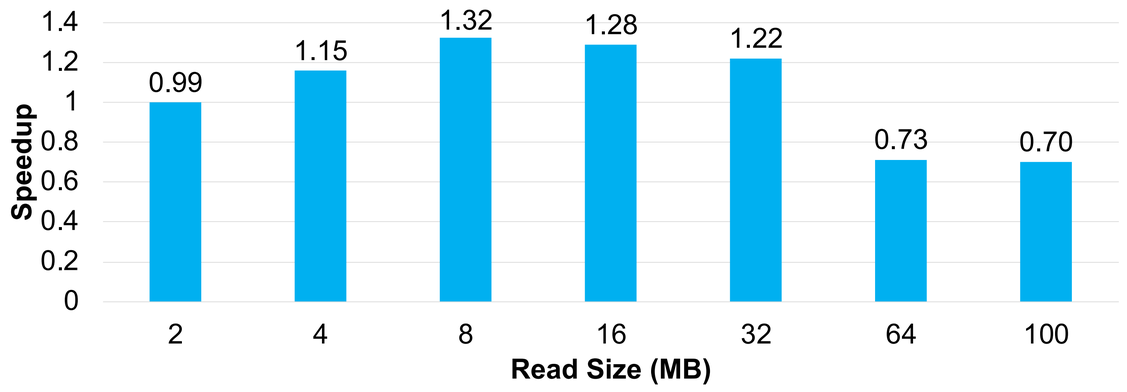
Speedup in adjusting data resolutions

MLSSD hides the latency of altering input datasets with accessing flash chips while taking advantages of richer internal parallelism, MLSSD accelerates the process of reading data inputs by 1.55x.



Shuffle mechanism1: Using PRP list

Shuffle mechanisms minimize memory copies compared to convention shuffle. Using PRP list, it shows a speedup of 1.23x when the read size is 1MB.



Shuffle mechanism2: Using Extents

The result shows that by partition requests into 8 MB chunks, MLSSD can speedup reads by 1.32x, in addition to the performance gain from moving data adjustments into the SSD.