**Review of Exploiting CXL-based Memory for Distributed Deep Learning**

**Motivation**

Deep learning workloads have large memory and storage requirements that typically exceed the limited main memory available on an HPC server. This increases the overall training time as the input training data and model parameters are frequently swapped to slower storage tiers during the training process.

**Contribution**

* This paper uses the latest advancements in the memory subsystem, specifically Compute Express Link (CXL), to provide additional memory and fast scratch space for DL workloads to reduce the overall training time while enabling DL jobs to efficiently train models using data that is much larger than the installed system memory.
* This paper implements and integrates DeepMemoryDL with a popular DL platform, TensorFlow, to show that our approach reduces read and write latencies, improves the overall I/O throughput, and reduces the training time.
* The evaluation shows a performance improvement of up to 34% and 27% compared to the default TensorFlow platform and CXL-based memory expansion approaches, respectively.

**Design Objects**

* Enable access to the additional memory space over the CXL interface for DL workloads
* Avoid throughput bottlenecks in the I/O pipelines and ensure minimal IO response time with prefetching intelligently and placing data close to the processing threads
* Provide fast CXL-based scratch space to store the training data, thus enabling high bandwidth access to data and eliminating IO access to slower storage tiers

**Design Overview**

A diagram of a workload

Description automatically generated

* **Resource Gatherer Module** that collects compute, memory, network, and storage resources of all servers included in the cluster.

A text in a dictionary

Description automatically generated

* **Workload Analyzer Module** that analyzes the submitted DL workload and breaks down the job in I/O and compute phases.

A text on a page

Description automatically generated

A text on a page

Description automatically generated

* **Core Module** has two parts: manager and workers. The manager resides on the same node as the master node in TF and supervises all operations of DeepMemoryDL.

A diagram of a software system

Description automatically generated

A close-up of a text

Description automatically generated

A black text on a white background

Description automatically generated

* **Prefetcher Module** that prefetches data and loads it in the main memory before it is required by the processing thread to minimize I/O stalls.

A close up of a text

Description automatically generated

A diagram of a memory storage system

Description automatically generated

* **Caching Policies**

A text on a page

Description automatically generated

A diagram of a workflow

Description automatically generated

**Result**

A graph of different colored bars

Description automatically generated

A graph of different colored lines

Description automatically generated