Sentinel: Efficient Tensor Migration and Allocation on Heterogeneous Memory Systems for Deep Learning

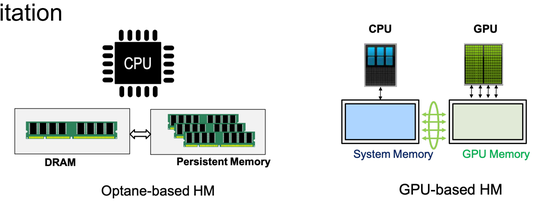
Jie Ren[1], Jiaolin Luo[1], Kai Wu[1], Minjia Zhang[2], Hyeran Jeon[1], Dong Li[1]

[1]University of California Merced [2]Microsoft

Motivation

Memory capacity becomes a major bottleneck for DL training

❑ Heterogenous Memory (HM) is Promising to Address Memory Capacity Limitation

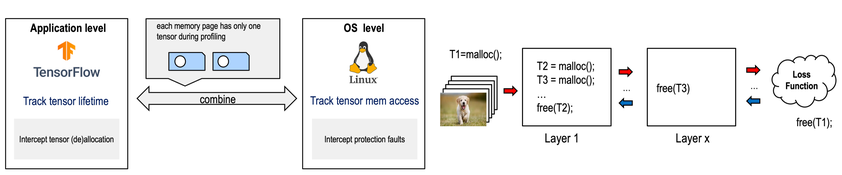


❑ State-of-the-art Tensor Management Solutions on HM

	Static Analysis DL Topology	Compiler-level profiling	Application-level Profiling
Example solutions	vDNN [1]	AutoTM [2]	Capuchin [3]
Graph-agnostic	✗	✓	✓
Dynamic profiling	✗	✗	✓
Avoid page-level false sharing	✗	✗	✗
Minimize fast mem usage	✗	✗	✗

Characterization of Main Memory Access in DNN

❑ Methodology



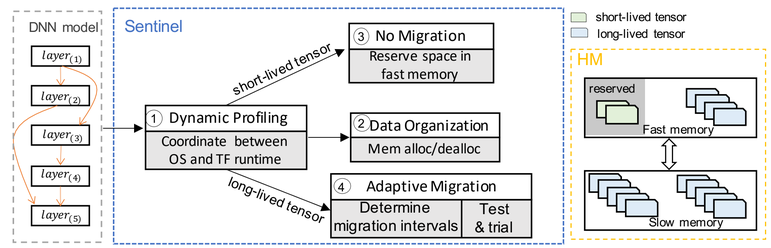
❑ Observations

- There are a large number of small tensors (i.e., smaller than 4KB) with short lifetime (i.e., lifetime smaller than one layer) in DNN training workloads.
- The uneven distribution of hot and cold tensors provides opportunities for tensor management.
- Page-level false sharing exists in DNN.
  - The page-level profiling (not tensor-level) for tensor management can be misleading.

Design

Sentinel, a software runtime system that automatically optimizes tensor management on HM.

❑ Overview



❑ Dynamic Profiling

- Profiling overhead is easily amortized over training steps .
  - Domain knowledge: exploiting workload repeatability of deep learning training.
- Memory overhead is ignorable.

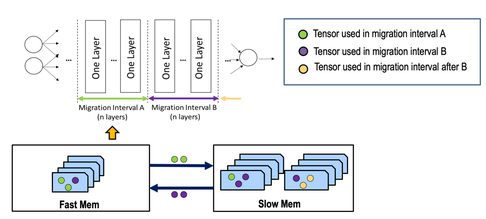
❑ Data Reorganization

- Sentinel reorganizes memory allocation for tensors to avoid page-level false sharing
  - Tensors with same lifetime and same access pattern will be placed in the same memory page to avoid page-level false sharing

❑ Lazy (No) Migration for Short-lived Tensors

- Reserve space in fast memory for short-lived tensors
  - Avoid unnecessary data movement
  - Reuse the same memory space to save fast memory space

❑ Proactive Migration for Long-lived Tensors



- The tensor migration is overlapped with DNN training computation as much as possible
  - Domain knowledge: The deep learning model topology (i.e., layers) and its depth to decide the optimal migration interval and trigger data migration.

Experiment Results

Sentinel consistently outperforms five state-of-the-art solutions on both Optane-based and GPU based training platform.

- Testing bed

Optane-based HM

Software platform:
Linux v4.9
TensorFlow v1.14

Hardware platform:
2-socket Intel Xeon Gold 6252 CPU@2.3GHz
DRAM: 96GB x 2; Optane: 756GB x2
The size of DRAM is equal to 20% of peak memory consumption of DNN models

GPU-based HM

Software platform:
HostOS: Linux v5.6.0
CUDA v10.1
TensorFlow v1.14

Hardware platform:
GPU: Nvidia V100 with 15.75 GB of GDDR6
CPU Intel(R) Xeon(R) E5-2670 with 128 GB of DDR4

- Overall Performance on Optane-based HM

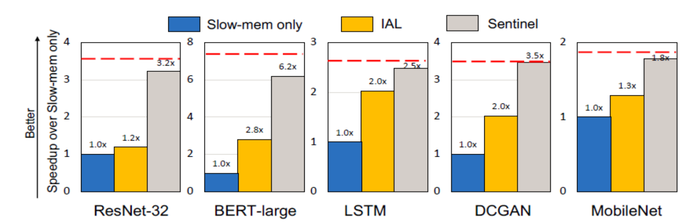
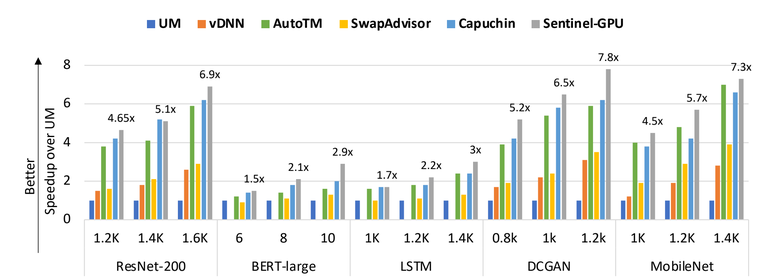


Fig. Performance speedup of IAL, Sentinel over slow memory only. The red horizontal line shows performance of fast-memory only. Performance is normalized by that of slow-memory only

✓ Performance difference between Sentinel and the DRAM-only system is very small

- Overall Performance on GPU-based HM



✓ Sentinel consistently outperforms five state-of-the-art solutions on GPU for models with various training batch size.

References

[1] M. Rhu, N. Gimelshein, J. Clemons, A. Zulfiqar, and S. W. Keckler, vDNN: Virtualized deep neural networks for scalable, memory-efficient neural network design, in MICRO, 2016
[2] M. Hildebrand, J. Khan, S. Trika, J. Lowe-Power, and V. Akella, "Autotm: Automatic tensor movement in heterogeneous memory systems using integer linear programming", in ASPLOS, 2020
[3] X. Peng, X. Shi, H. Dai, H. Jin, W. Ma, Q. Xiong, F. Yang, and X. Qian, "Capuchin: Tensor-based gpu memory management for deep learning,", in ASPLOS, 2020