Can Unified Memory support on Pascal and Volta GPUs enable Out-of-Core DNN Training?

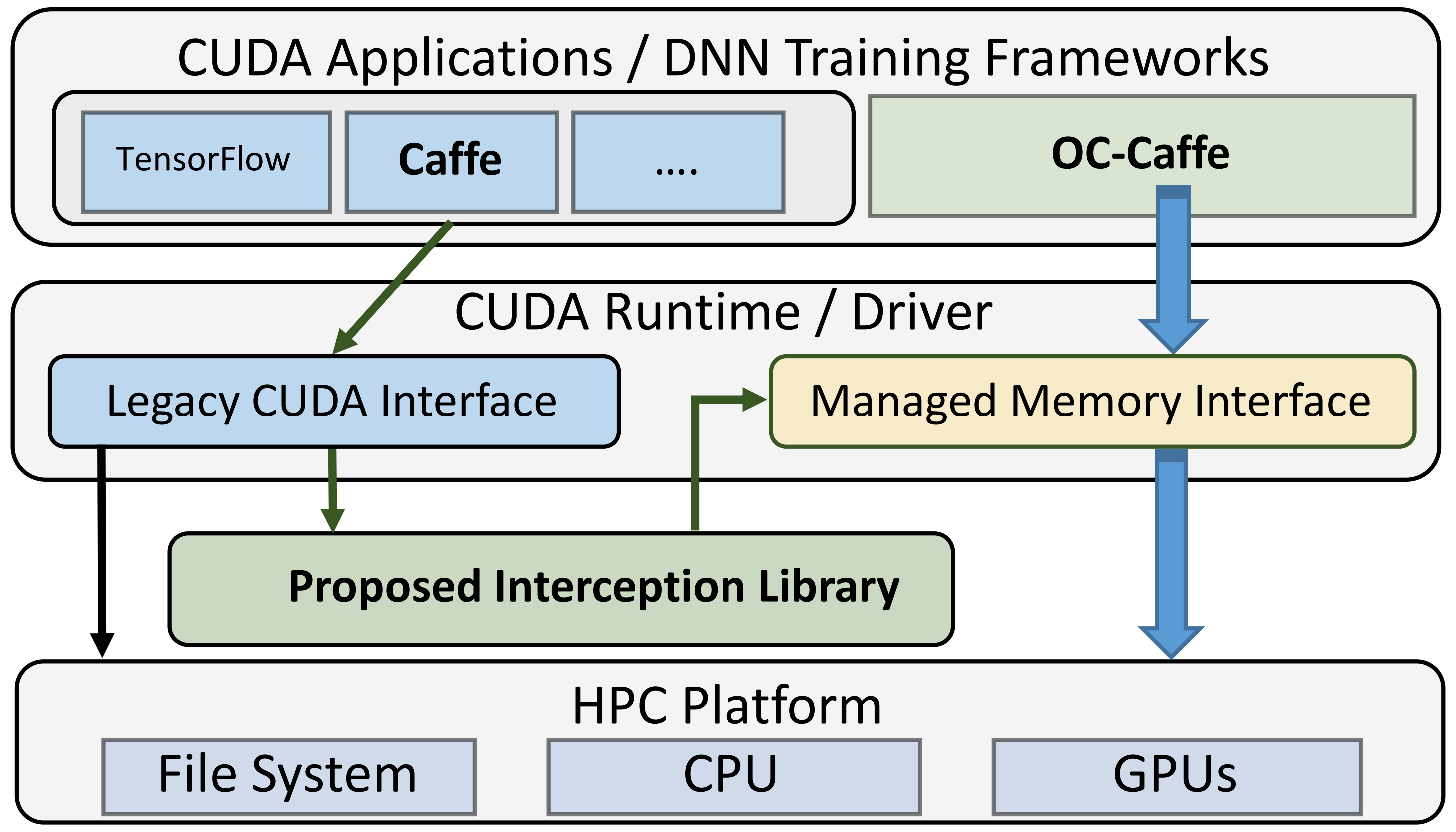
MOTIVATION

- Resurgence of Deep Learning (DL)
  - Availability of Large Datasets like ImageNet and massively-parallel modern hardware like NVIDIA GPUs
  - Emergence of DL frameworks (Caffe, TensorFlow, CNTK, etc.)
- Existing DL frameworks cannot train large Deep Neural Networks (DNNs) and/or large batch sizes for certain DNNs
  - GPU memory is limited so larger models/batch sizes do not fit
  - How to design Out-of-core support in DL frameworks?
- New Unified-Memory (UM) features in CUDA 8/9 and enhanced support in Pascal/Volta GPUs
  - Investigate CUDA UM for Out-of-core DNN training

RESEARCH CHALLENGES

- Can we decompose DNN training operations into fundamental CUDA-level primitives?
- How to deal with large amount of training data?
- How to efficiently tackle intra-GPU communication for out-of-core DNN training?
- What are the alternatives for out-of-core training?

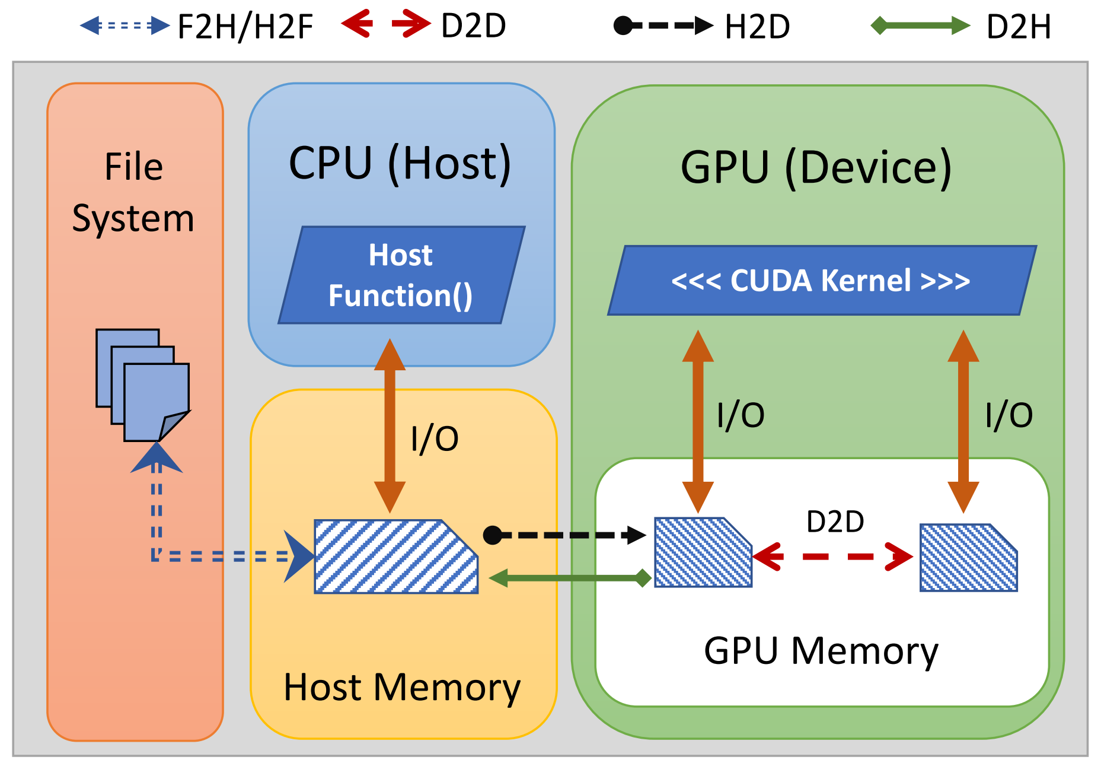
PROPOSED FRAMEWORK



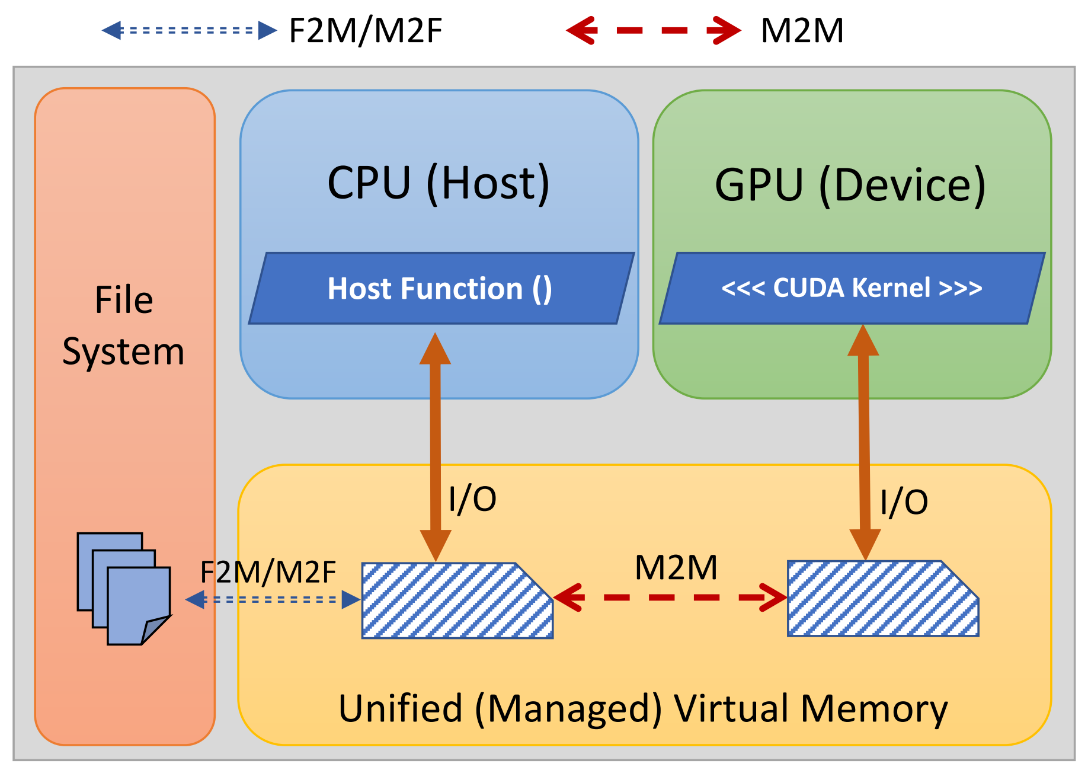
SUMMARY OF CONTRIBUTIONS

- **O**ut-of-**C**ore **D**eep **N**eural **N**etwork (OC-DNN) framework for efficient out-of-core DNN training on a single GPU by exploiting managed-memory primitives.
- Several design schemes for OC-Caffe to illustrate the applicability of the proposed OC-DNN framework and how managed-memory primitives can be exploited for out-of-core DNN training.
- Productivity and performance benefits for training prevalent DNNs like ResNet-50, VGG, GoogLeNet, and AlexNet on cutting edge GPU architectures like Pascal and Volta.
- Design scale-up and scale-out designs in OC-Caffe for distributed DNN training on multiple GPUs

PROPOSED PRIMITIVES

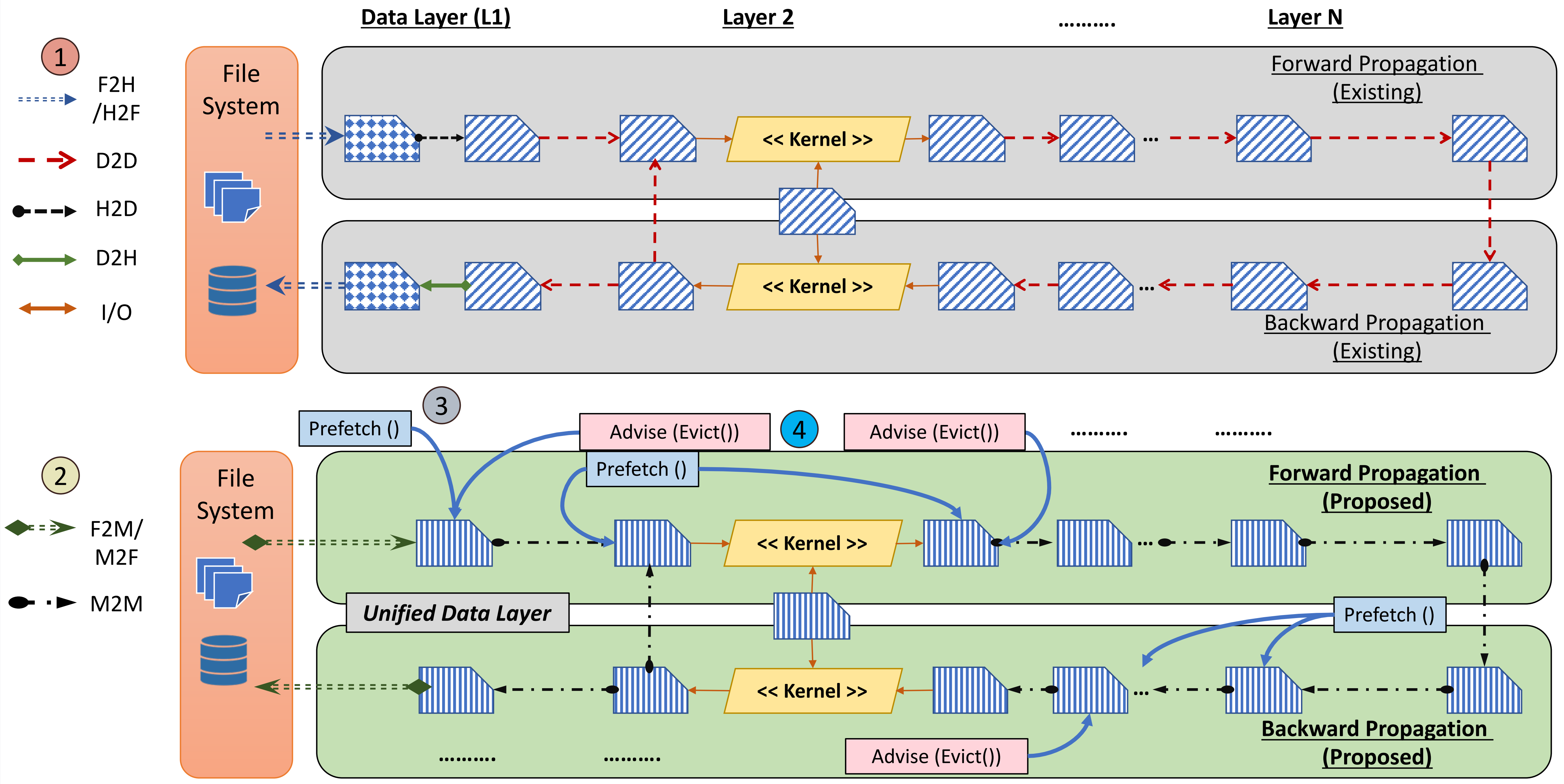


Existing Primitives using Explicit CPU and GPU Allocation (malloc, cudaMalloc, and cudaMallocHost)



Proposed Primitives using Unified Memory Allocations (cudaMallocManaged)

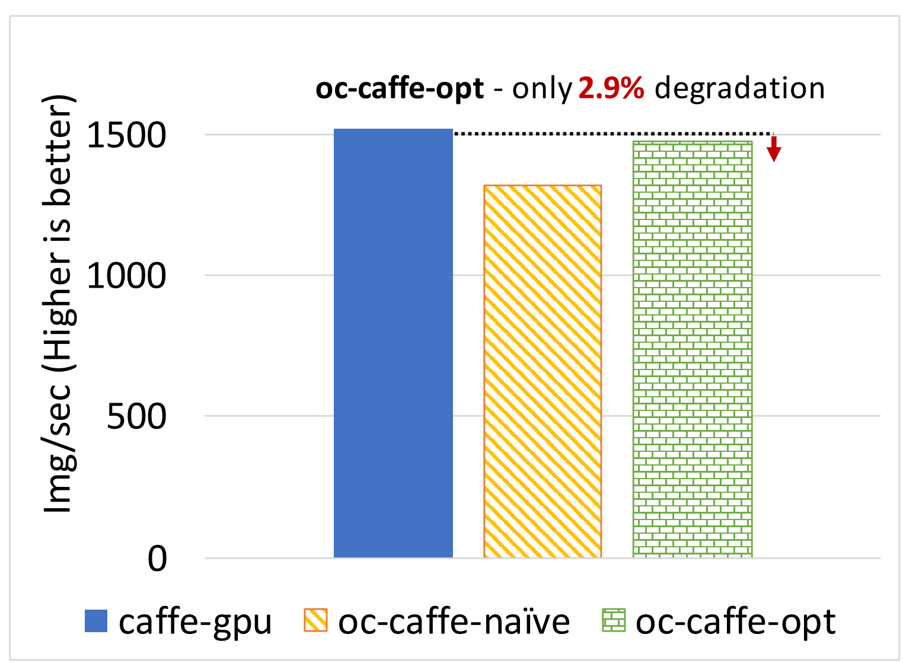
PROPOSED OC-CAFFE DESIGN



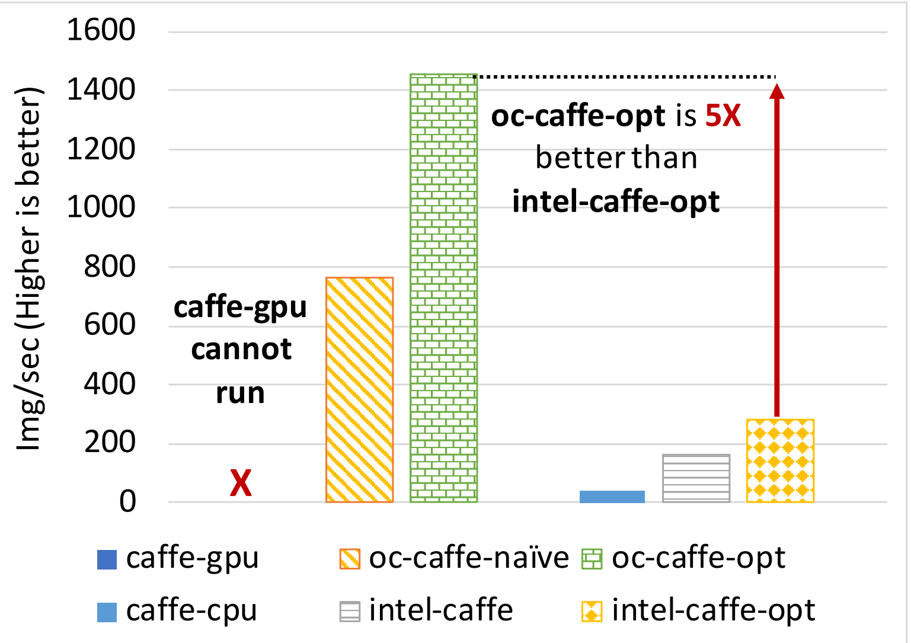
- ***Optimizing File Access in OC-Caffe***
  - Exploit the proposed unified-memory primitives like F2M/M2F (1) instead of F2H/H2F (2)
- ***Optimizing Intra-GPU Communication for Faster Training***
  - Leverage cudaMemPrefetch (3) and cudaMemAdvise calls (4)
  - Prefetch and Evict data in an on-demand manner to minimize page faults and get better performance

PERFORMANCE BENEFITS

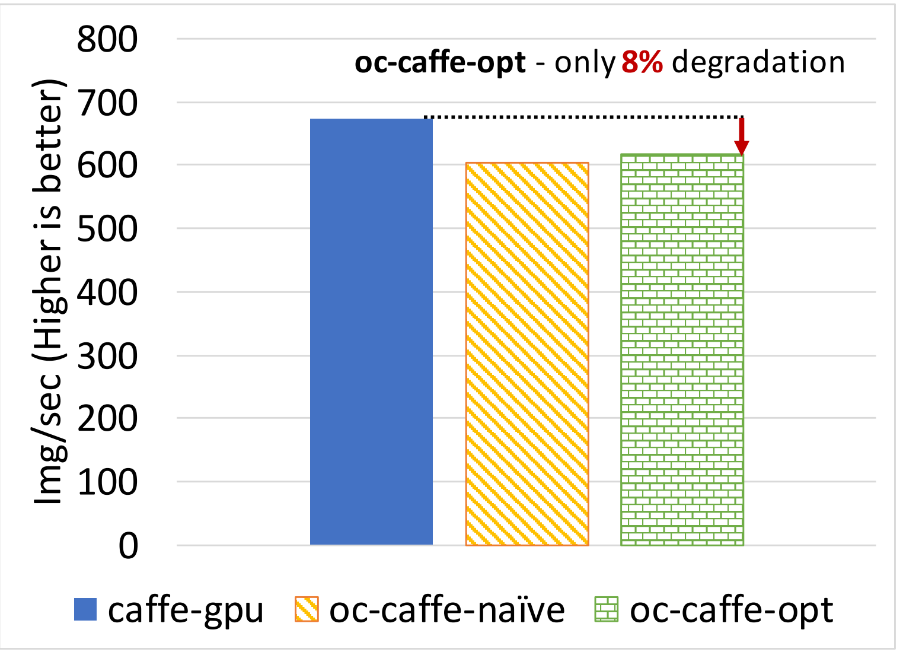
Trainable AlexNet



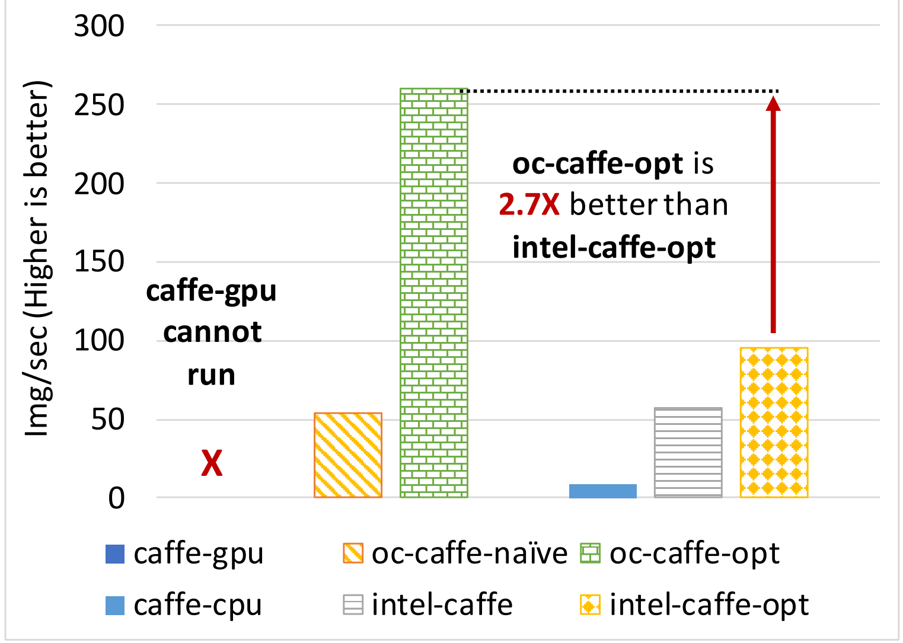
Out-of-core AlexNet



Trainable GoogLetNet



Out-of-core GoogLeNet



← ← AlexNet and GoogLeNet Training Performance

- ***OC-Caffe*** only minor degradation compared to ***Caffe-Default*** for **"trainable"** batch sizes
- ***OC-Caffe-Optimized*** design provide up to ***19% improvement*** over Naïve and up to ***5X better*** than CPU-based training for **"out-of-core"** training

Simpler Design with OC-Caffe →→

- Remove significant memory allocation, movement, and state-management code
- Estimated 3,000 lines of repetitive and error-prone code can be eliminated
- Simplify Layer implementations in ***OC-Caffe***

PRODUCTIVITY BENEFITS

Existing Design

```
class ConvolutionLayer
{
public:
   void cpu_data()
   void cpu_diff()
   void gpu_data()
   void gpu_diff()

   void mutable_cpu_data()
   void mutable_cpu_diff()
   void mutable_gpu_data()
   void mutable_gpu_diff()

   void Forward_cpu()
   void Forward_gpu()
   void forward_cpu_gemm()
   void forward_gpu_gemm()
   void forward_cpu_bias()
   void forward_gpu_bias()

   void Backward_cpu()
   void Backward_gpu()
   void backward_cpu_gemm()
   void backward_gpu_gemm()
   void backward_cpu_bias()
   void backward_gpu_bias()
}
```

Proposed High-Productivity Design based on Managed Memory Allocation and Data Movement

```
class ConvolutionLayer
{
public:
   void data()
   void diff()

   void mutable_data()
   void mutable_diff()

   void Forward()
   void forward_gemm()
   void forward_bias()

   void Backward()
   void backward_gemm()
   void backward_bias()
}
```