FreeLunch: Compression-based GPU Memory Management for Convolutional Neural Networks

Shaurya Patel  
University of British Columbia  
University of Massachusetts, Amherst

Tongping Liu  
University of Massachusetts, Amherst

Hui Guan  
University of Massachusetts, Amherst
CNN memory consumption trend

Memory Consumption

GB

Alexnet | VGG | ResNet50 | ResNext50 | ResNet152 | ResNet256 | GPipe5

56.5% | 71.5% | 76.1% | 77% | 78.3% | 81.8% | 84.6%
Forward Activations (much larger in size than model params) need to persist in memory until the gradient updates in backward phase!
Policies for memory management

• Swapping
  • Capuchin [X. Peng et al., 2020]
  • SwapAdvisor [C-C. Huang et al., 2020]
  • Superneurons [L. Wang et al., 2018]
  • …

• Recomputation
  • Capuchin [X. Peng et al., 2020]
  • Superneurons [L. Wang et al., 2018]
  • …
CPU-GPU bandwidth is a bottleneck!

OOM!

Swapping

CPU
Recomputation is complex and has lineage dependencies!
• A compression based policy for CNN training.
  • basic Idea: **compress and keep the tensors on GPU memory.**
  • avoids the bandwidth issue introduced by swapping.
  • avoids the computation complexity of recomputation.

• Challenges:
  • How to reduce the compression overhead?

  Parallel workflow

Optimizations:
  • Sliding Compression Workspace
  • Persistent Tensor Buffers
Parallel workflow

GPU

Training cudaStream1

Compression queue

FreeLunch cudaStream2

C1 → C2 → C3 → C4
Typical Compression workflow

Memory operations synchronize all cuda streams!

cudaMalloc()  cudaMalloc()  cudaFree()  cudaMemcpy()  cudaFree()
This workflow introduces multiple blocking operations!
Persistent tensor buffers

**GPU**

- **params**
- **ACT4**
- **ACT2**
- **ACT3**

**Training cudaStream1**

1. C1 → C2 → C3 → C4

**Compression queue**

**FreeLunch cudaStream2**
• Can FreeLunch improve training throughput while reducing memory consumption of CNN training?

• How effective are the optimizations in FreeLunch compared with other compression-based baselines?
Throughput as compared to other policies

- Liveness
- Swapping
- Recomputation
- FreeLunch

32% vs 70%
Memory consumed as compared to other policies
No observed impact on accuracy of model
Performance comparison between FreeLunch and [Jin et al.]

Impact of optimizations

<table>
<thead>
<tr>
<th>Model</th>
<th>Throughput (images/second)</th>
<th>Optimization Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>25</td>
<td>1.69X</td>
</tr>
<tr>
<td>ResNet152</td>
<td>15</td>
<td>1.85X</td>
</tr>
<tr>
<td>ResNet256</td>
<td>5</td>
<td>1.92X</td>
</tr>
</tbody>
</table>

- No optimizations
- FreeLunch
• Capuchin and SwapAdvisor use swapping in an async manner.
• We implement async swapping and compare it to FreeLunch.

Throughput comparison with async swapping

Async swapping vs FreeLunch

<table>
<thead>
<tr>
<th>Model</th>
<th>Throughput (images/second)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>1.32X</td>
</tr>
<tr>
<td>VGG16</td>
<td>1.32X</td>
</tr>
<tr>
<td>ResNet256</td>
<td>1.13X</td>
</tr>
</tbody>
</table>

Throughput comparison with async swapping

Async Swapping vs FreeLunch
We implemented a hybrid async swapping policy in combination with FreeLunch.
Summary

• We introduce FreeLunch that effectively avoids the bandwidth and concurrent execution that swapping and recomputation face.

• We incorporate two optimizations as part of FreeLunch to make compression parallelizable and improve performance.

• We show that FreeLunch achieves up to 70% better throughput and up to 32% better memory consumption.