Towards GPU Utilization Prediction for Cloud Deep Learning

Gingfung Yeung, Damian Borowiec, Adrian Friday, Richard Harper, Peter Garraghan

Evolving Distributed System Lab
School of Computing & Communications
Lancaster University
UK
Deep Learning (DL) Systems

Machine Learning engineers, researchers, users + More Deep Learning (DL) workloads + Growing number of expensive GPUs

Require efficient resource usage & high DL performance
DL System Challenges

• Avg. GPU utilization
  ~ 52% in production systems [Jeon et al. '19]

• Long job completion + queue times
  ~ up to hours [Jeon et al. '19; Gu et al. '19]

Addressed via understanding and exploiting workload patterns
Online profiling approach

Deploy workload into isolated machines and GPUs to obtain workload patterns.

- **Workload**
  - Deployed into isolated machines and GPUs to obtain workload patterns.

- **Node**
  - Resource Monitor
  - Node
  - Profile
  - Response

- **Workload**
  - GPU-1 (Utilization = 20, Memory = 4GiB,Bytes…)
  - GPU-2 (Utilization = 40, Memory = 6GiB,Bytes…)

- **Usual per workload profiling range from minutes to hours**

- **Comments**
  - Usually per workload profiling range from minutes to hours.
DL Metrics

- **Iteration time**
  - Useful for scale-out workers, migration, SLA-aware inference
  - [Peng et al. ’18; Xiao et al.’ 18; Shen et al.’ 19]

- **Network I/O**
  - Useful for efficient distributed training
  - [Gu et al. ’19]

- **GPU Utilization**
  - For packing and calculating interference
  - [Thinakaran et al. ’19; Xu et al. ’19]
Case: Scheduling

1. Query
2. Issue
3. Migrate

Make decision based on workload patterns from profiling

Scheduler

Resource Monitor

Resource Management Framework

Scheduling Loop

```func Schedule {
  Profile()
  CheckMigrate()
  // ....
  // ....
  // ....
}```
Time is Money

- $N$ workload $\times$ mins

If the system has many heterogenous workloads, will lead to head-of-line blocking.
Online Profiling

• **Pros**
  • Accurate, near real-time workload patterns
  • Provide insights to the system

• **Cons**
  • Heterogenous workloads require different profiles
  • Time consuming (~mins to ~hours)
  • Require modifying underlying frameworks
Online Profiling

- **Pros**
  - Accurate, near real-time workload patterns
  - Provide insights to the system

- **Cons**
  - Heterogenous workloads require different profiles
  - Time consuming (~mins to ~hours)
  - Require actual execution onto an isolated machine
  - Require modifying underlying frameworks

Obtain prior execution?
Prediction

- $N$ workload $\times$ seconds

reduce blocking

Workload Queue

Scheduling Stage

Prediction Stage (sub-second – seconds)
DL System Challenges

• Avg. GPU utilization
  ~ 52% in production systems [Jeon et al. ’19]

• Long job completion + queue times
  ~ up to hours [Jeon et al. ’19; Gu et al. ’19]

Addressed via understanding and exploiting workload patterns
DL Metrics

- **Iteration time**
  - Useful for scale-out workers, migration, SLA-aware inference
  - [Peng et al. '18; Xiao et al.’ 18; Shen et al.’ 19]

- **Network I/O**
  - Useful for efficient distributed training
  - [Gu et al. ’19]

- **GPU Utilization**
  - For packing and calculating interference
  - [Thinakaran et al. ’19; Xu et al. ’19]
Objective

GPU utilization prediction engine for Cloud DL Systems

Benefits

• Estimates GPU utilization of unseen workloads
• Prior to execution
• No modification of existing DL frameworks
  • E.g. PyTorch, TensorFlow, MXNet…

Analysis, prediction model, case study
Going deeper with convolutions [Szegedy et al 2014]

Leverage graph information to predict workload usage.

\[ f(x) \rightarrow y \]

Features: Num. Convs, FLOPs, layers, etc. (See paper for full features list)
Analysis

- **Profile DL workload utilization**
  - Determine important model features

- **Set up**
  - Nvidia 1080, Nvidia 2080, Intel i7-6850k
  - 13 DNN model architectures, 81 workloads
  
  See paper for full list of models and permutations.

- **Tools**
  - Nvidia-smi
  - Nvidia Nsight Systems
Analysis
Analysis

**1.5x – 4x slowdown from co-location**
GPU Utilization Prediction

\[
\sqrt{\frac{1}{n} \sum_{i=1}^{n} (\log(p_i + 1) - \log(y_i + 1))^2}
\]

<table>
<thead>
<tr>
<th>Model</th>
<th>RMSLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>0.291</td>
</tr>
<tr>
<td>LightGBM</td>
<td>0.255</td>
</tr>
<tr>
<td>XGBoost</td>
<td>0.197</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.154</td>
</tr>
</tbody>
</table>
Evaluation

33.5% Makespan reduction
61.5% Utilization improvements
Open Challenges

• Hardware
  • Number of processing elements, memory bandwidth and cache sizes.

• DL Compilers
  • Extract lower level IR to determine optimization decision for more accurate prediction. (e.g. Op fusion – ConvBatchNorm)

• Distributed Workload
  • Network I/O, parallelism strategy and system configuration.
    • (e.g. ring topology)

• Co-location Scheduling
  • Incorporate prediction and system constraints
  • Derive an optimization algorithm
    • (e.g. Mixed Integer Programming).