Lancaster University

Towards GPU Utilization Prediction for Cloud Deep Learning

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

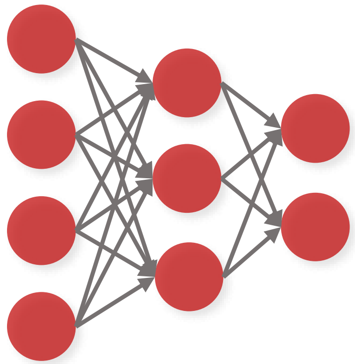
Evolving Distributed System Lab
School of Computing & Communications
Lancaster University
UK

DS-LAB
THE DISTRIBUTED SYSTEMS LAB
AT LANCASTER

Deep Learning (DL) Systems

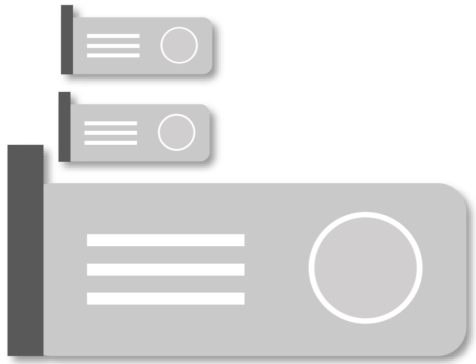
Machine Learning engineers, researchers, users

+



More Deep Learning (DL) workloads

+

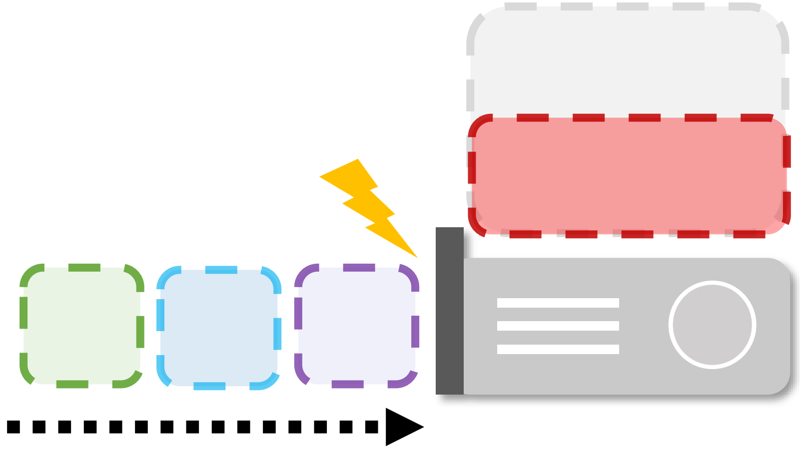


Growing number of expensive GPUs

Require efficient resource usage & high DL performance

DL System Challenges

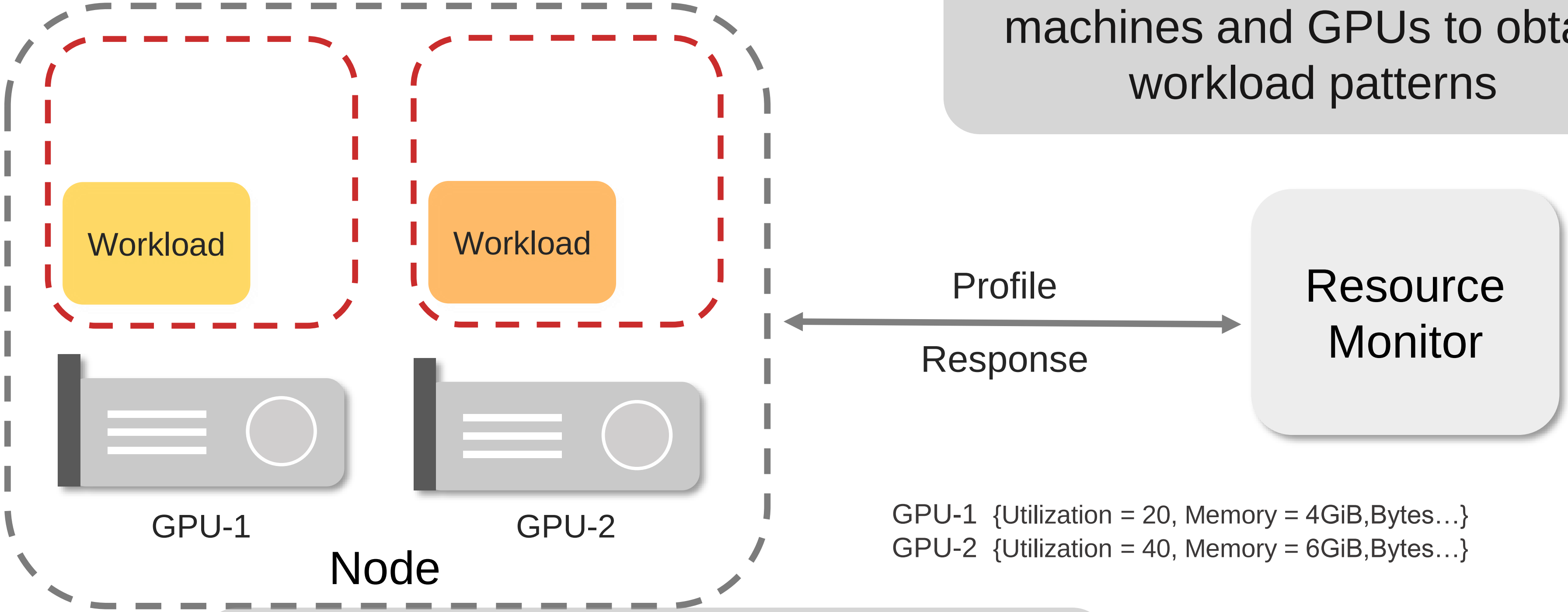
- Avg. GPU utilization
  - ~ 52% in production systems [Jeon et al. ’19]
- Long job completion + queue times
  - ~ up to hours [Jeon et al. ’19; Gu et al. ‘19]



Addressed via understanding and exploiting workload patterns

Online profiling approach

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

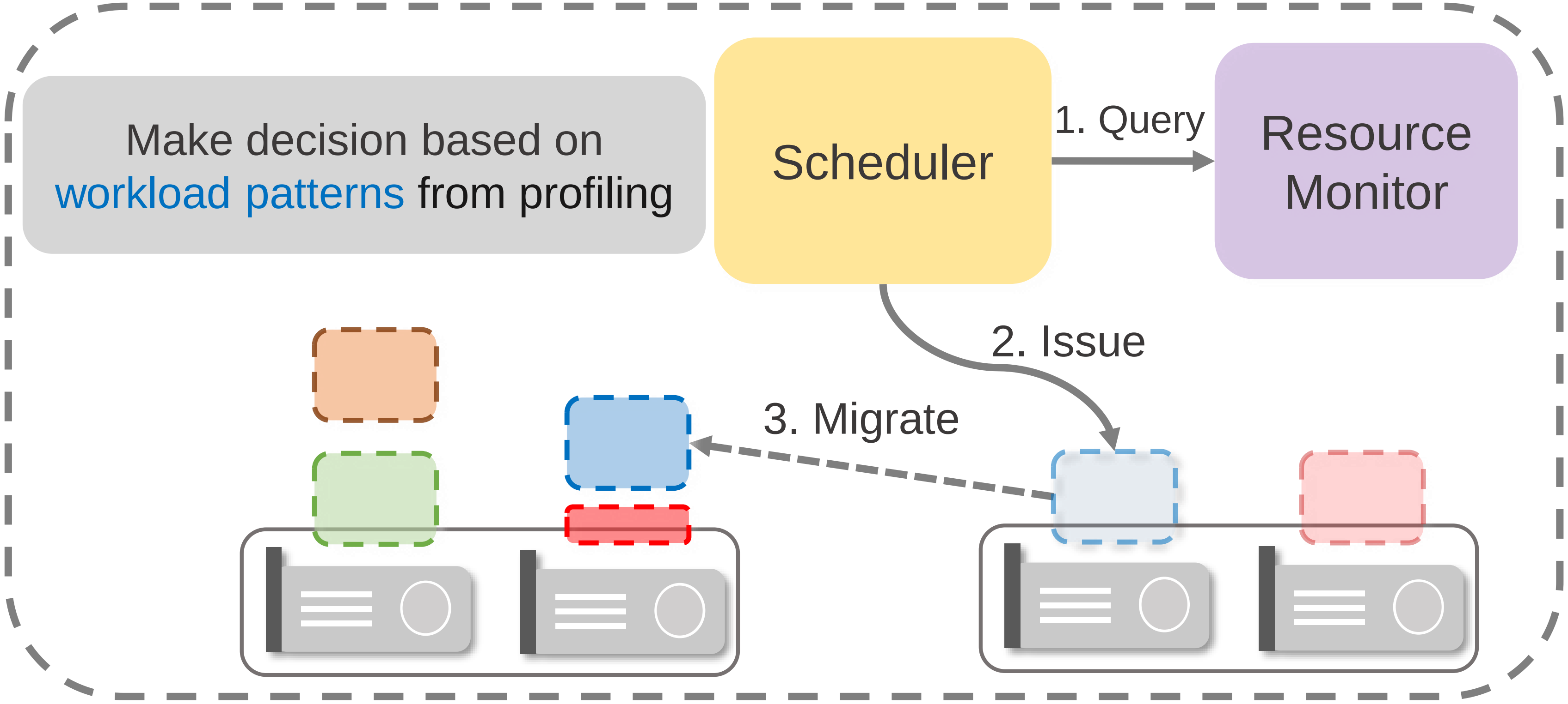


Usually per workload profiling range from **minutes** to **hours**

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



Resource Management Framework

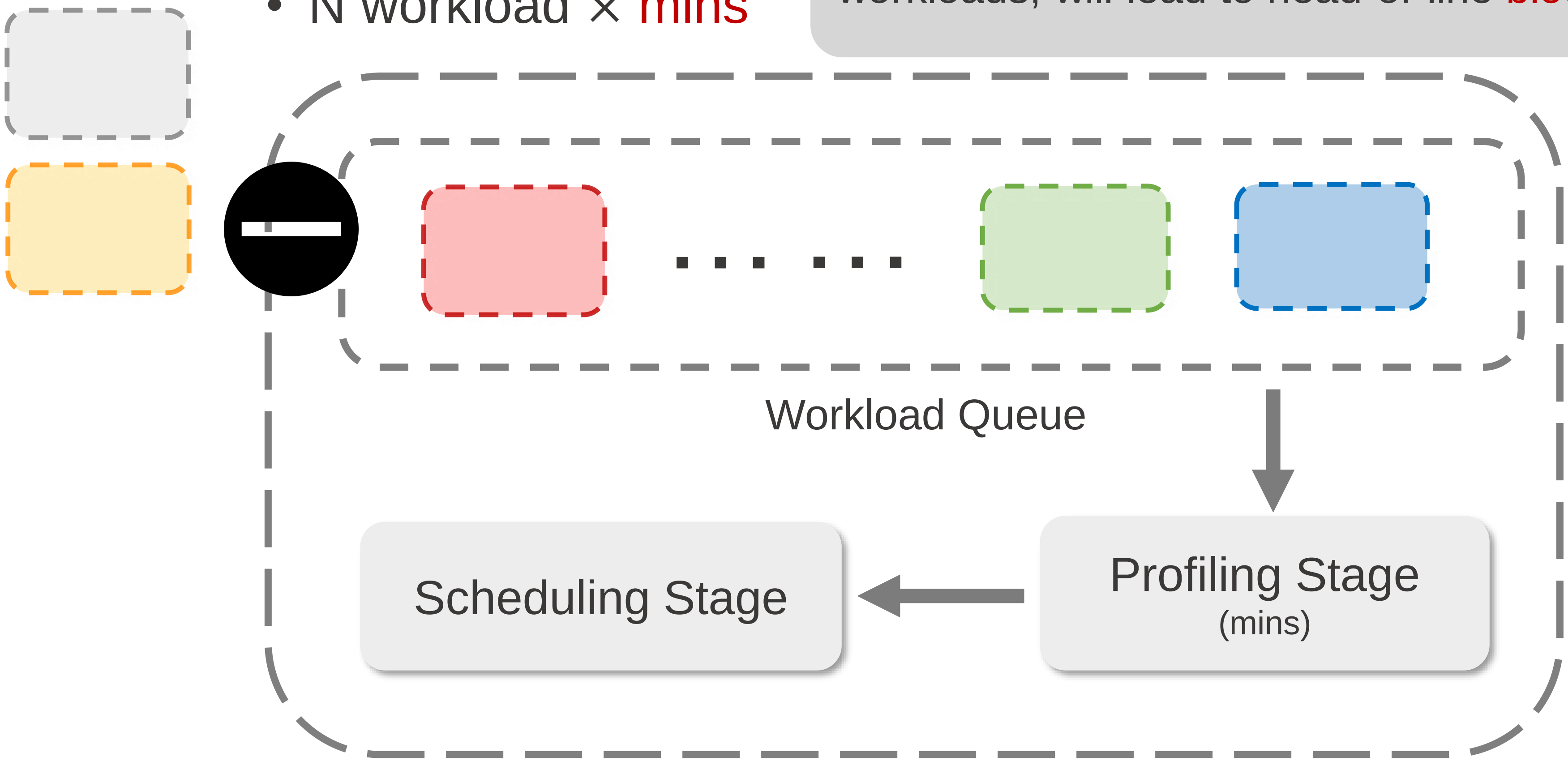
Scheduling Loop

```
func Schedule {
    Profile()
    CheckMigrate()
  // ....
  // ....
}
```

Time is Money

- N workload × 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**

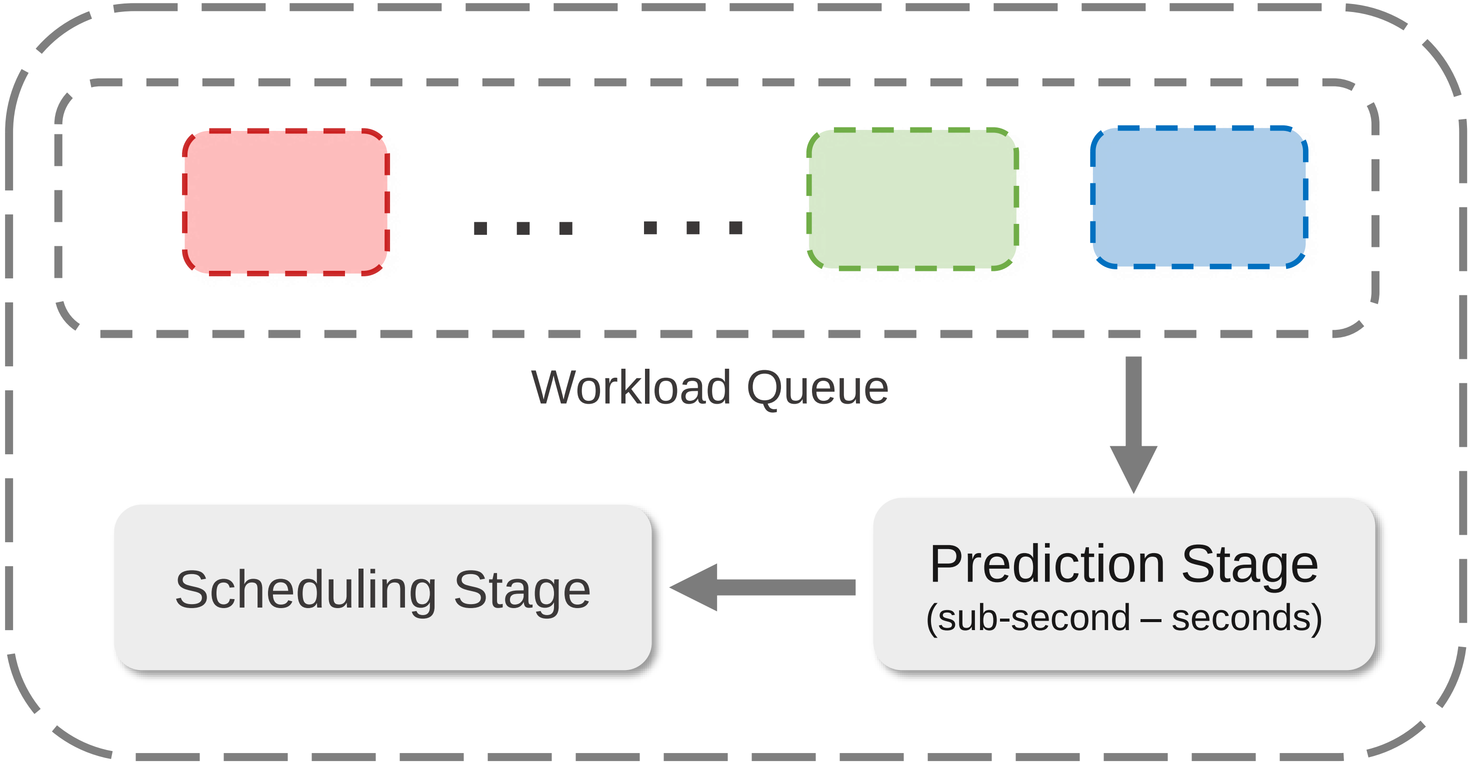
Obtain prior execution ?

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

Prediction

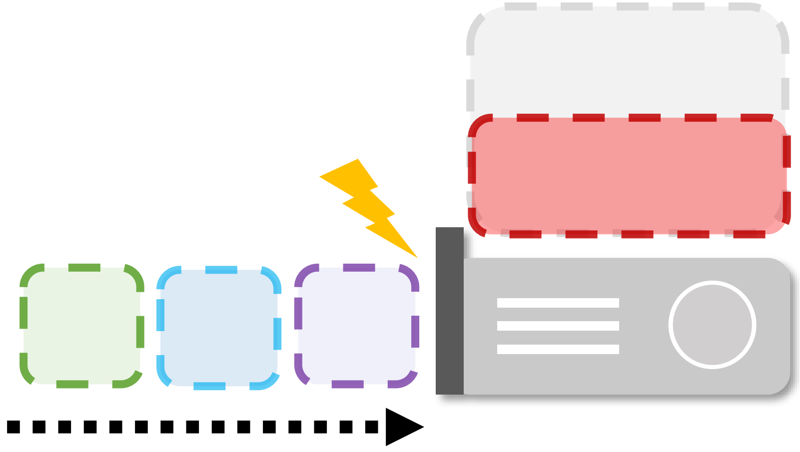
- N workload × seconds

Reduce blocking



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

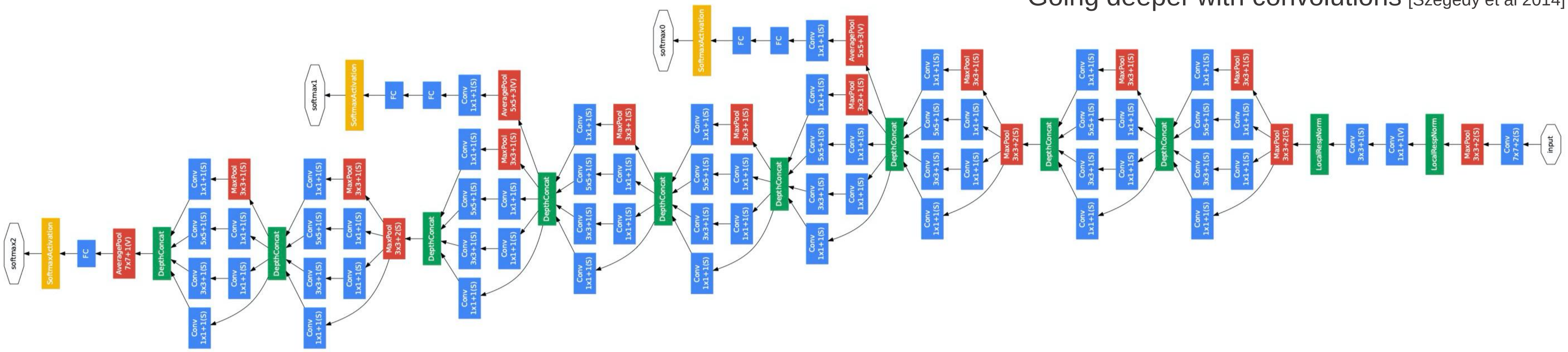
Benefits

- 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

DL computation graph

Going deeper with convolutions [Szegedy et al 2014]



Leverage graph information to **predict** workload usage.

$$f(x) \rightarrow y$$

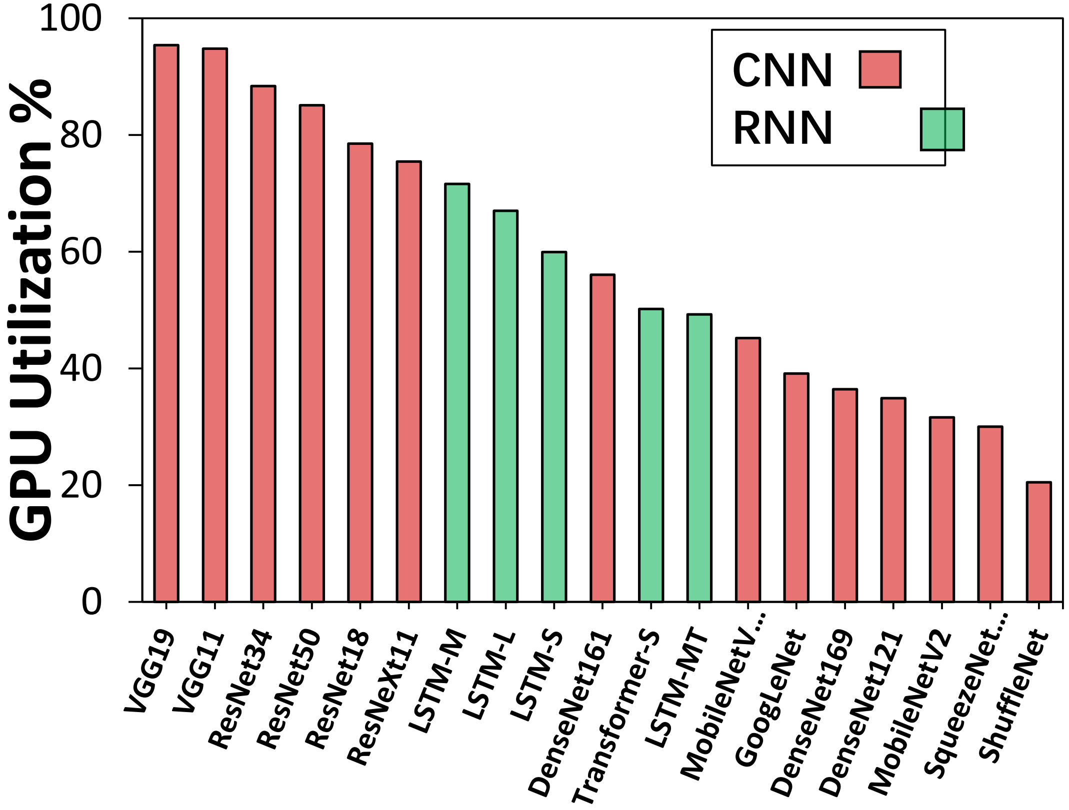
Features: Num. Convs, FLOPs, layers, etc.
(See paper for full features list)

Analysis

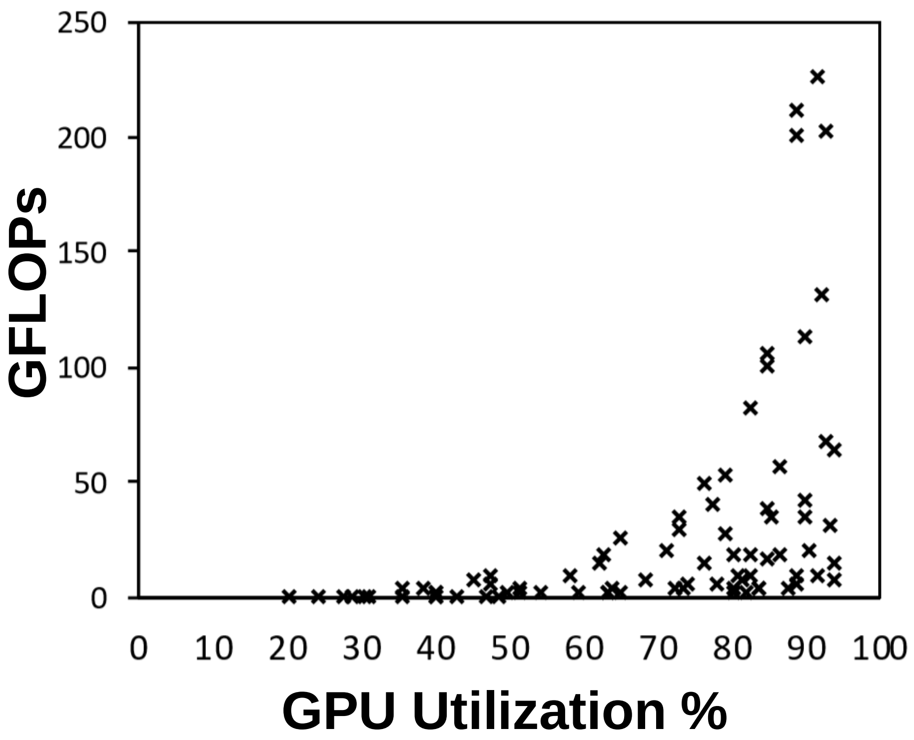
- **Profile DL workload utilization**
  - Determine important model features
- **Set up**
  - Nvidia 1080, Nvidia 2080, Intel i7-6850k
  - 13 DNN model architectures, 81 workloads

    See paper for full list of models and permutations.
- **Tools**
  - Nvidia-smi
  - Nvidia Nsight Systems

Analysis

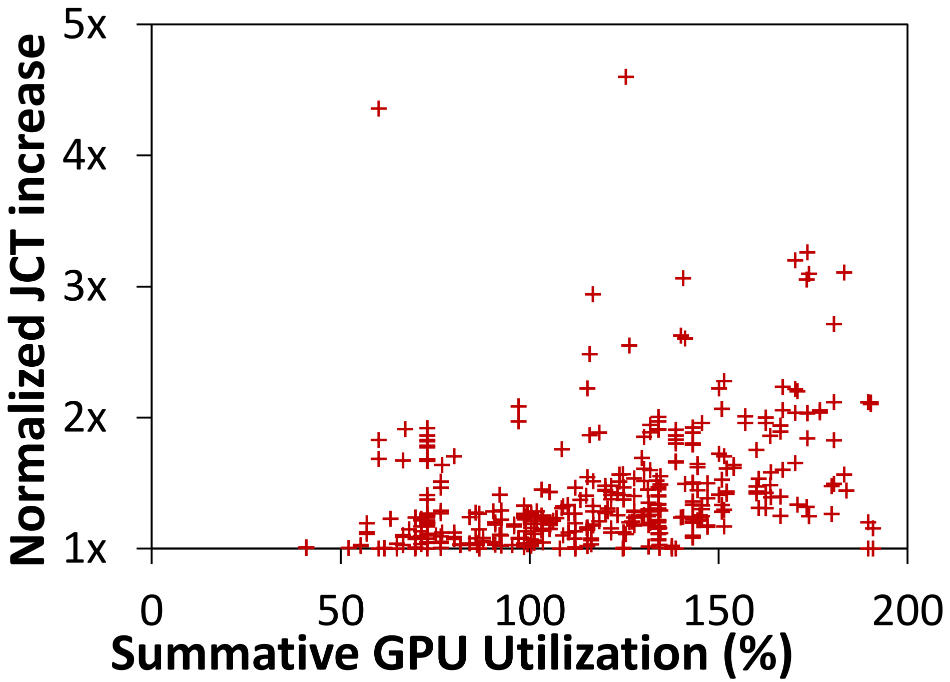


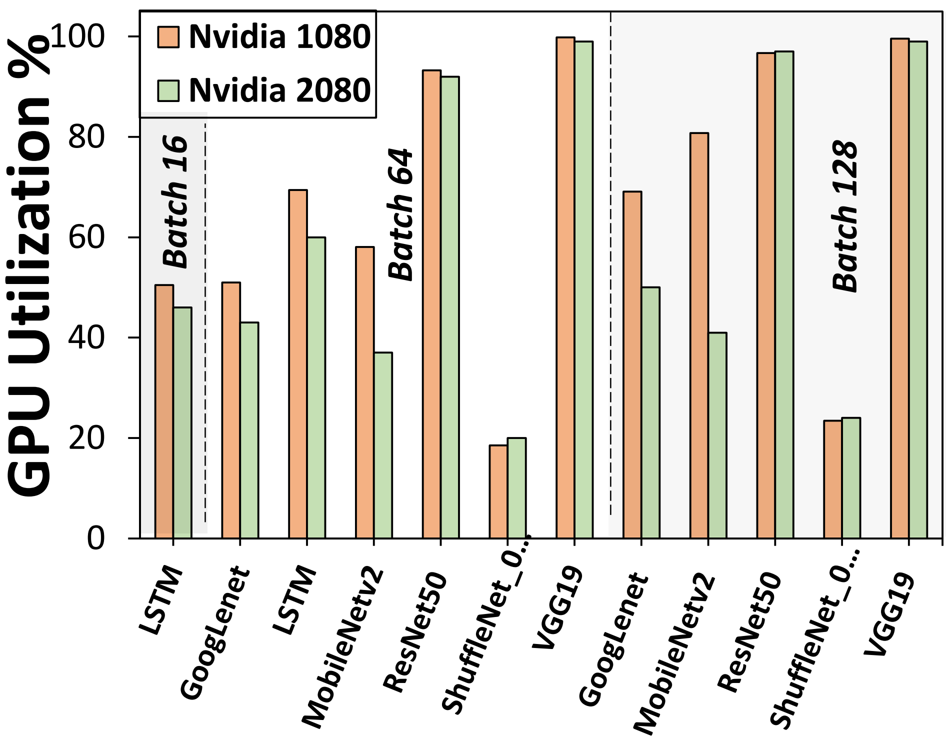
GPU Utilization %
100
80
60
40
20
0
CNN
RNN
VGG19
VGG11
ResNet34
ResNet50
ResNet18
ResNeXt11
LSTM-M
LSTM-L
LSTM-S
DenseNet161
Transformer-S
LSTM-MT
MobileNetV...
GoogLeNet
DenseNet169
DenseNet121
MobileNetV2
SqueezeNet...
ShuffleNet



GFLOPs
250
200
150
100
50
0
0 10 20 30 40 50 60 70 80 90 100
GPU Utilization %

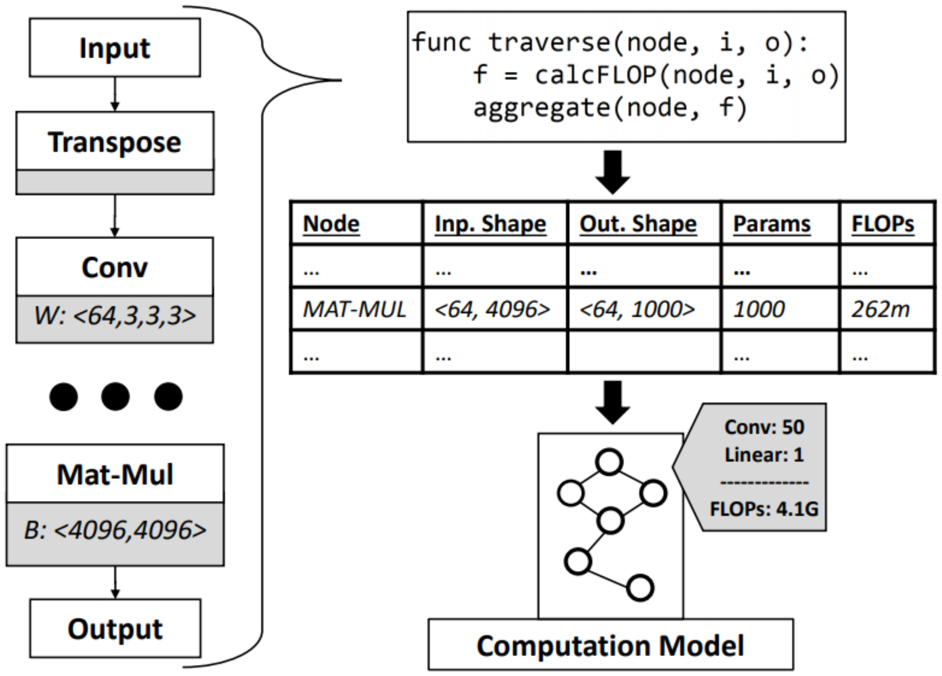
Analysis





1.5x – 4x slowdown from co-location

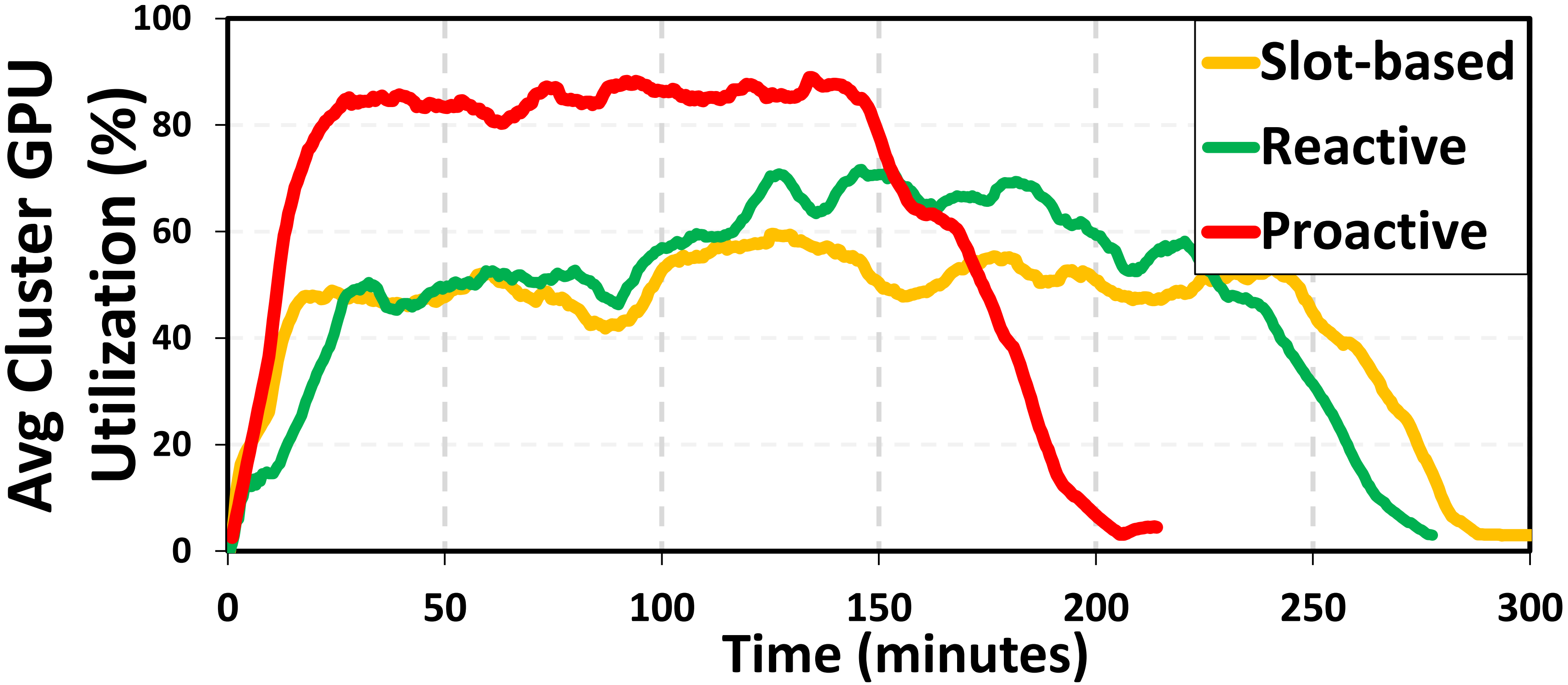
GPU Utilization Prediction



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

	RMSLE
Linear	0.291
LightGBM	0.255
XGBoost	0.197
Random Forest	0.154

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).