109學年度大學部專題競賽

國立清華大學資訊工程學系
Department of Computer Science, National Tsing Hua University

Optimizing Deep Learning Workloads on Kubernetes

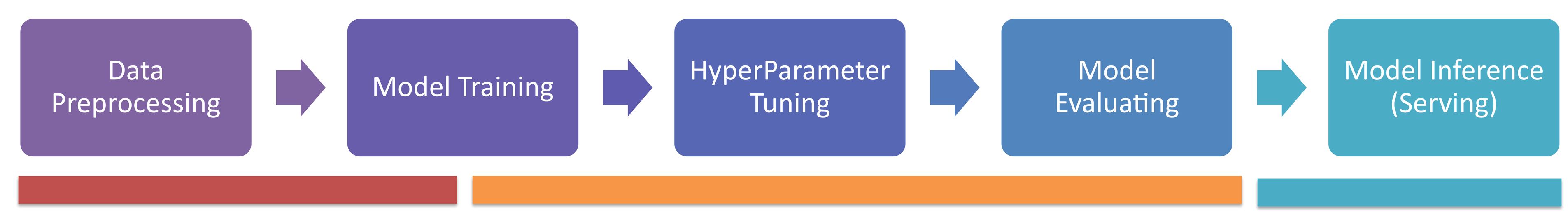
Mencher Chiang 江岷錡

Motivation

- ML / DL become increasingly popular workload
- Resource-Intensive workloads and Expensive GPUs
- ✅ We need to reduce the computational costs by
1. Maximize Resource Utilization
2. Minimize Resource Consumption of Each Workload
- Build a MLSys that optmizes the ML Workloads

Background

Apart from model training, a true ML workload is a ML Pipeline:

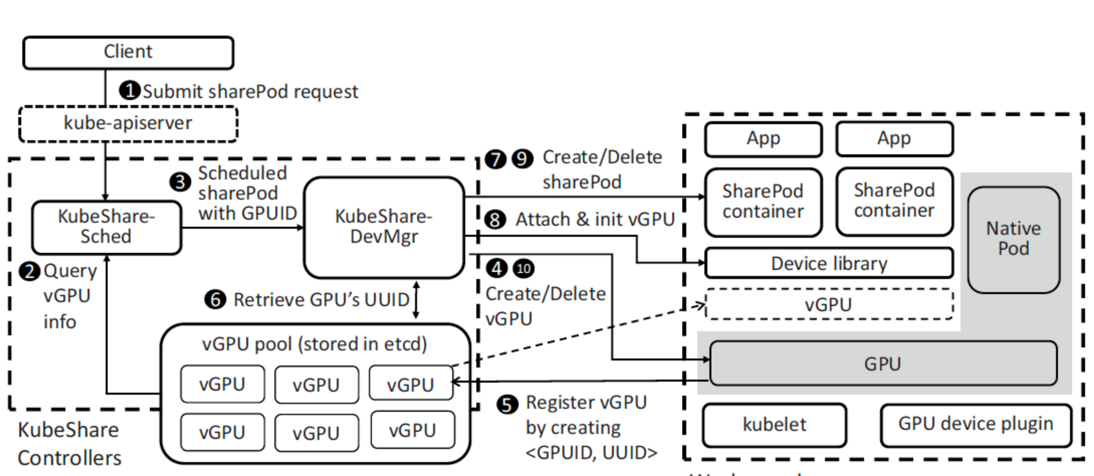


Three main workloads: Model Developing / Training / Inference

Proposed Methodology

Based on Kubernetes, the microservice system for containerized application
Cloud Native for DevOps, Easy to scale and portable, Extensible and well Supported

Model Developing
Kubeshare

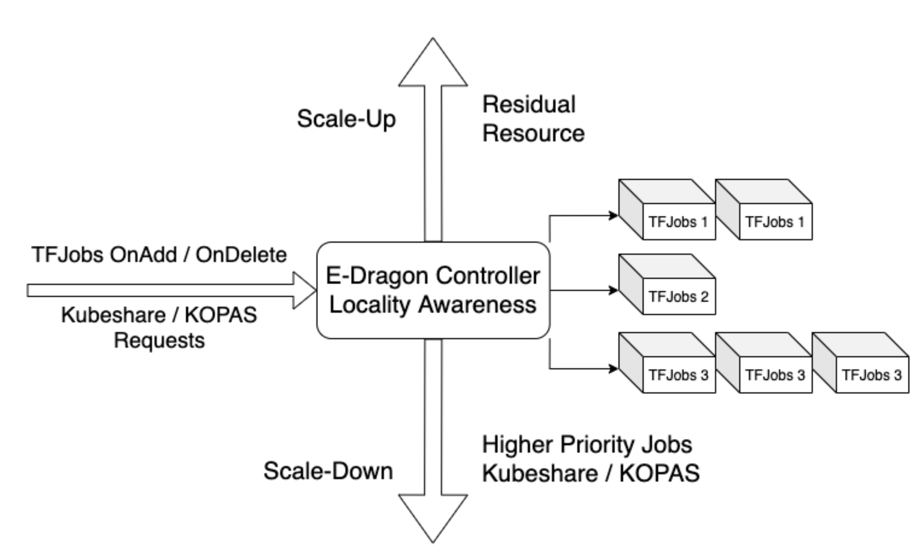


Low Resource Consumption and Sporadic jobs

Goal
- ✓ Less Waiting Time
- ✓ Avoid Resource Waste
- ✓ Still Keep Sufficient Performance

Technique
Consolidating GPU on multiple
Notebook instances by employing Kubeshare

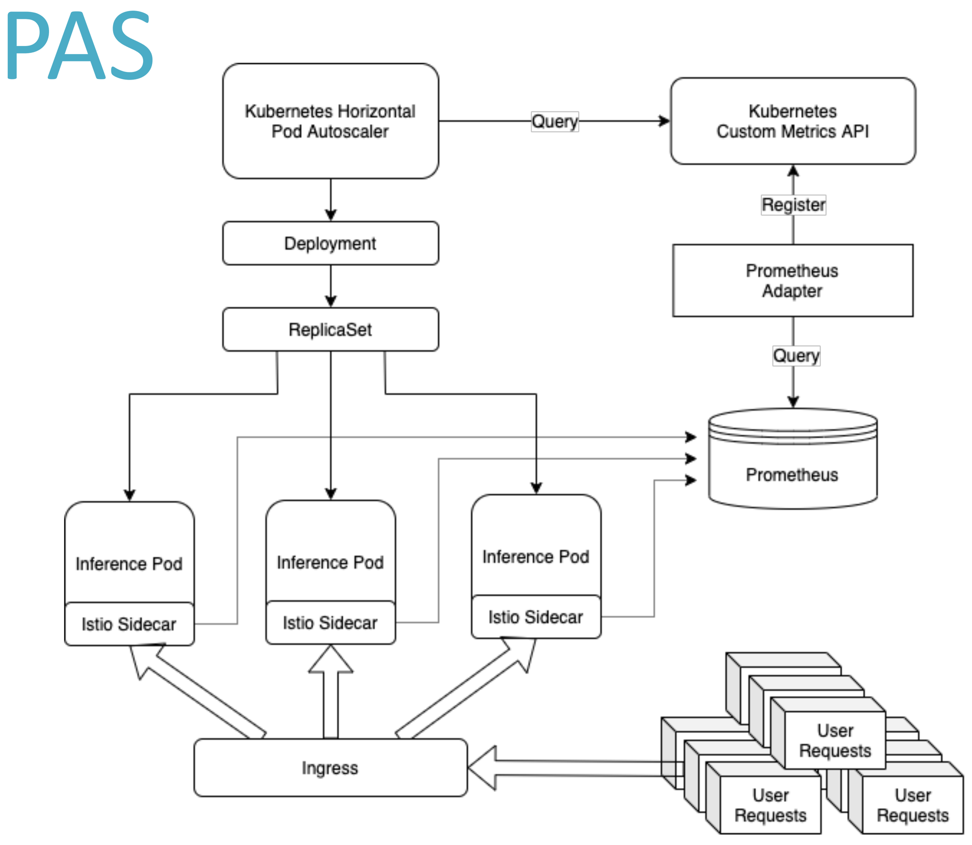
Model Training
Enhanced Dragon



Goal
- ✓ Improve Resource Utilization
- ✓ Reduce Training Time

Technique
Distributed Training
Gang-scheduling with Locality Awareness
Priority Scheduling

Model Inference
KOPAS

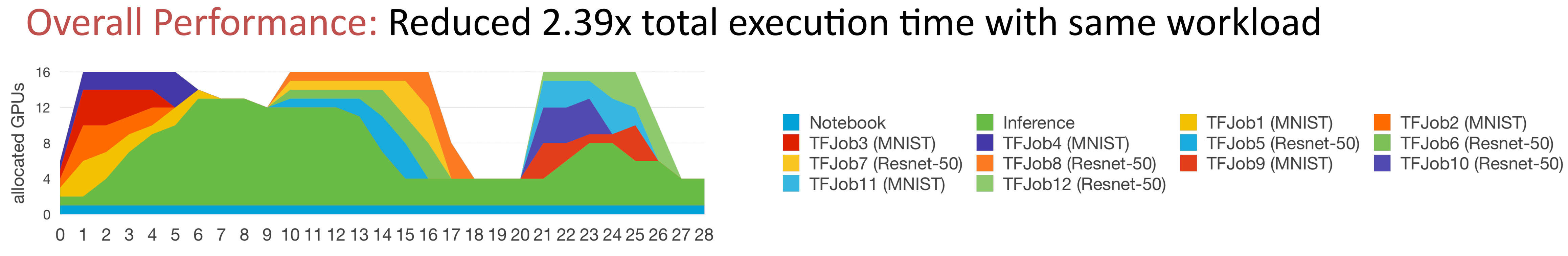


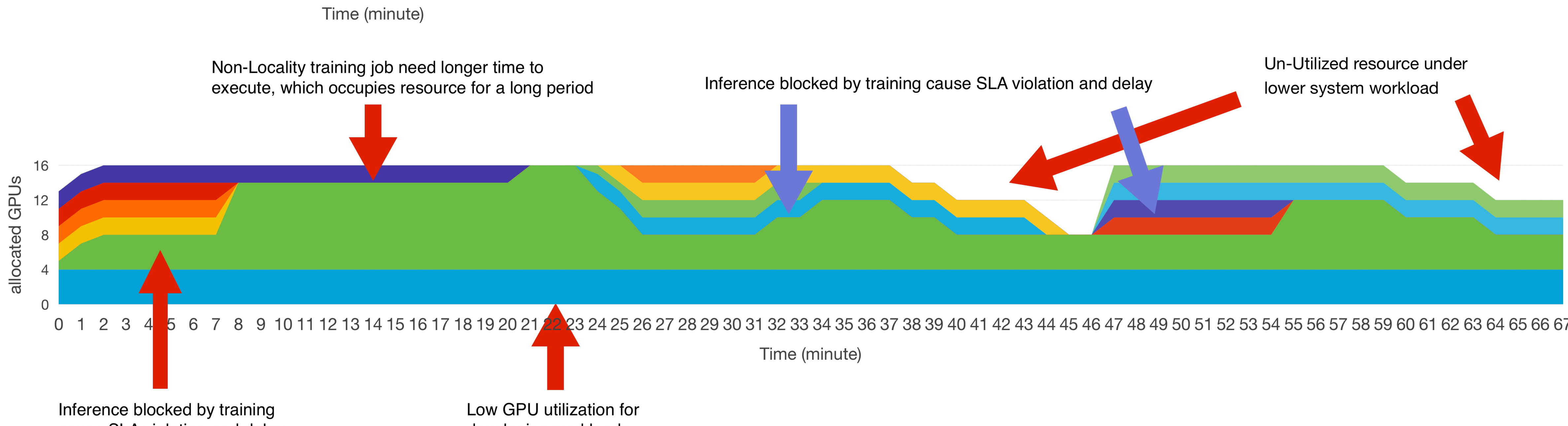
Goal
- ✓ Guarantee Service Quality
- ✓ Lower Response Time

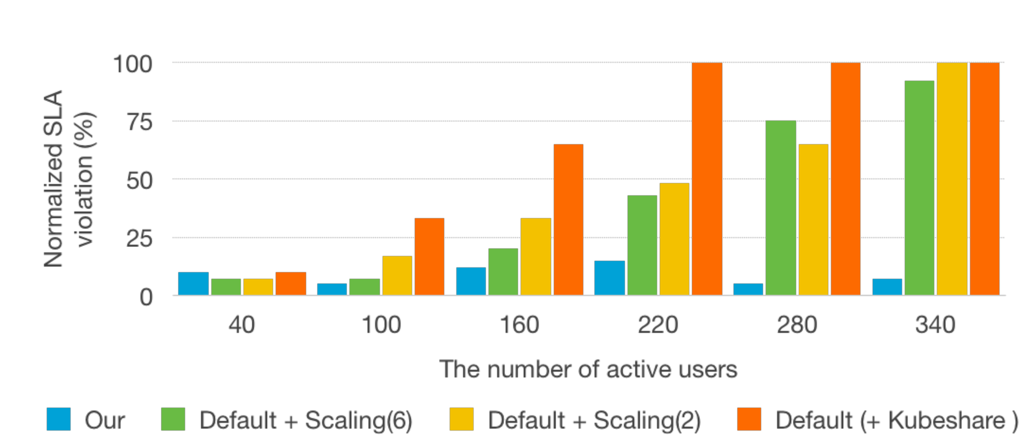
Technique
Kubernetes HPA + Istio + Prometheus
Priority Scheduling and dynamically scaling

Experimental Result

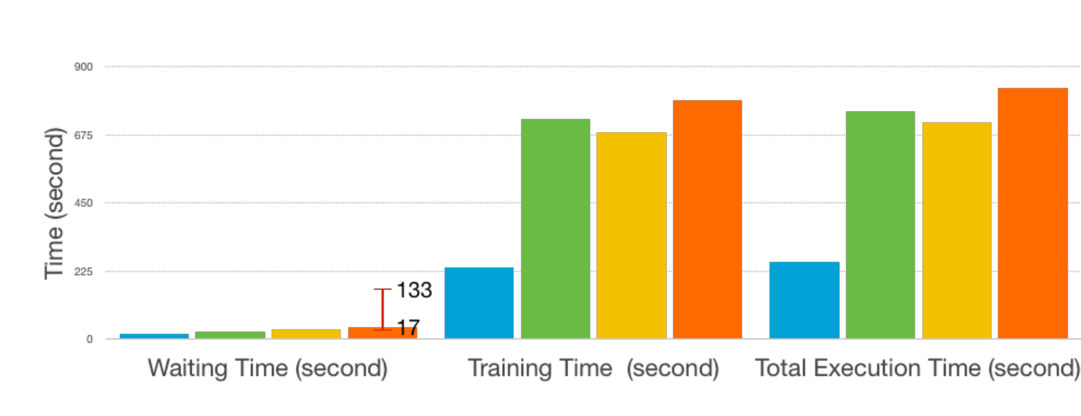
Overall Performance: Reduced 2.39x total execution time with same workload







Inference Job: Nearly No SLA Violation



Training: Reduced 3.26x Total Execution Time