Scavenger: A Cloud Service for Optimizing Cost and Performance of ML Training

Sahil Tyagi and Prateek Sharma
{styagi, prateeks}@iu.edu

Problem

- Given wide array of size and types of VMs available in the cloud, challenging to find right cluster configuration in the cloud
- Incorrect allocation either increases training time or cost of ML training.
- Is there a way to find correct cluster configuration?

Methodology

- Data-parallel training for K workers and batch B:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \frac{1}{K}\frac{1}{b}\sum_{k=1}^{k=K} \nabla f(\mathbf{x}_{k,t})$$

- Training **scales-up** by increasing B; **scales-out** by increasing K
- Not all work equally important as measured by Gradient Noise:

$$\gamma(t) = \frac{\mathbb{E}[\frac{1}{K}\sum_{k=1}^{K} ||g_t^{(k)}||^2]}{\mathbb{E}[||\tilde{g}_t||^2]}$$

- Time and cost predicted by knowing total iterations needed, per-step time and VM price:

$$T = n_i \cdot \tau_1 \text{ for } n_i = \frac{eD}{B} \text{ and } C = T \cdot K \cdot p$$

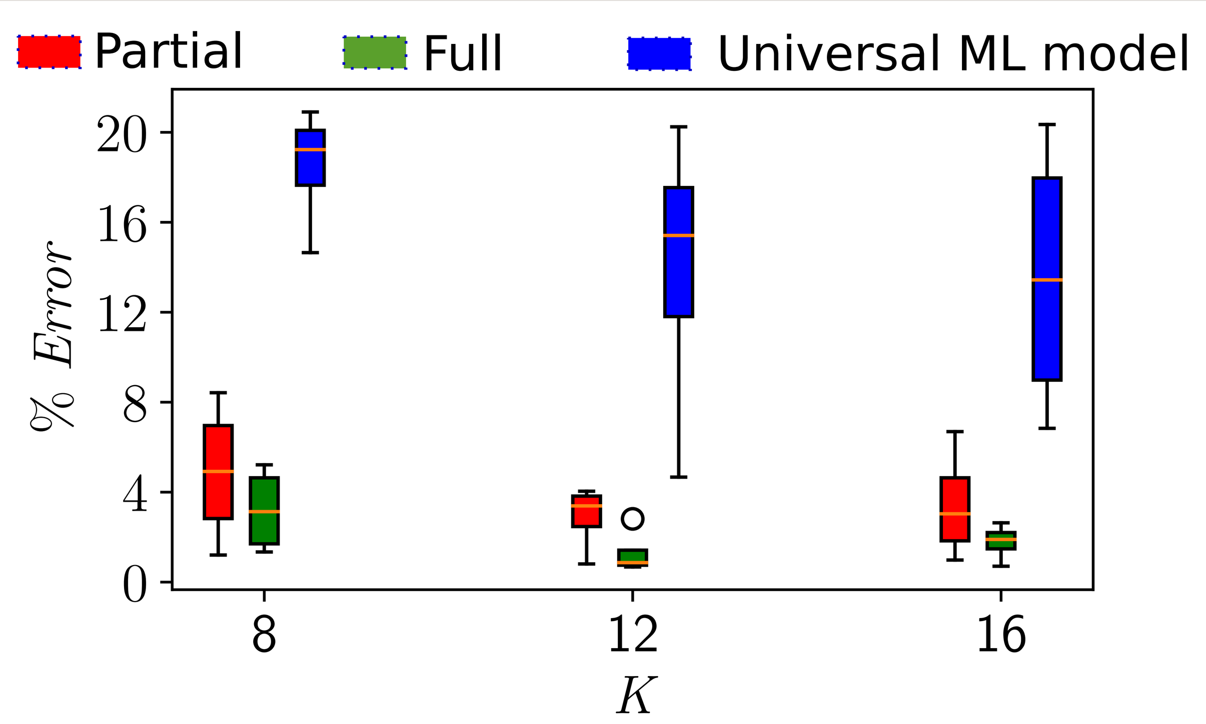
- **Statistical performance** modeled by the relationship $e \propto \gamma$ & $\gamma \propto 1/\sqrt{B}$
- **Parallel performance** models step time as:

$$t_{step} = t_{compute} + t_{sync}$$
where $t_{compute} \propto b$ and $t_{sync} \propto K$

- Performance modeling uses either **full-search, partial-search** or **no-search**.
- Predict time and cost from model and build tradeoff curves; select configuration based on user preference: **minimize time, minimize cost or knee-point**

Prediction error of search techniques

- Full search runs each configuration so **most accurate**; partial search runs at extreme points and interpolates; universal model averages all prior models so **least accurate**.
- **Error between 4-20% only** across all!

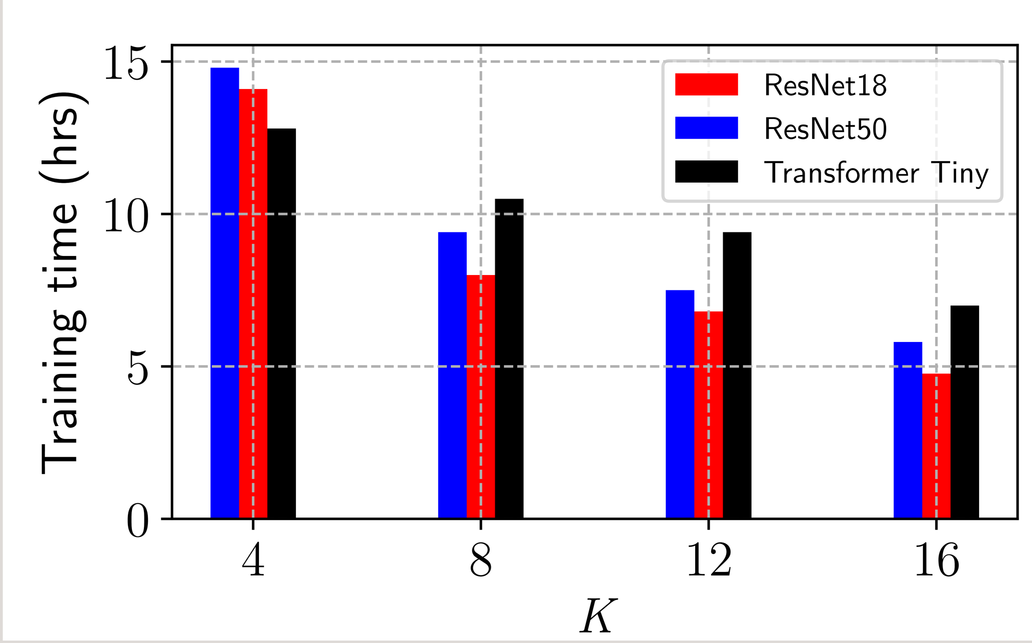


Contributions

- Scavenger finds ideal configuration reducing time by 2x!
- Online, black-box method that predicts time and cost of a training job with 98% accuracy
- Builds parallel and statistical performance models with minor overheads
- Build Gradient noise as scaling indictor for horizontal/vertical scaling

Horizontal scaling in the cloud

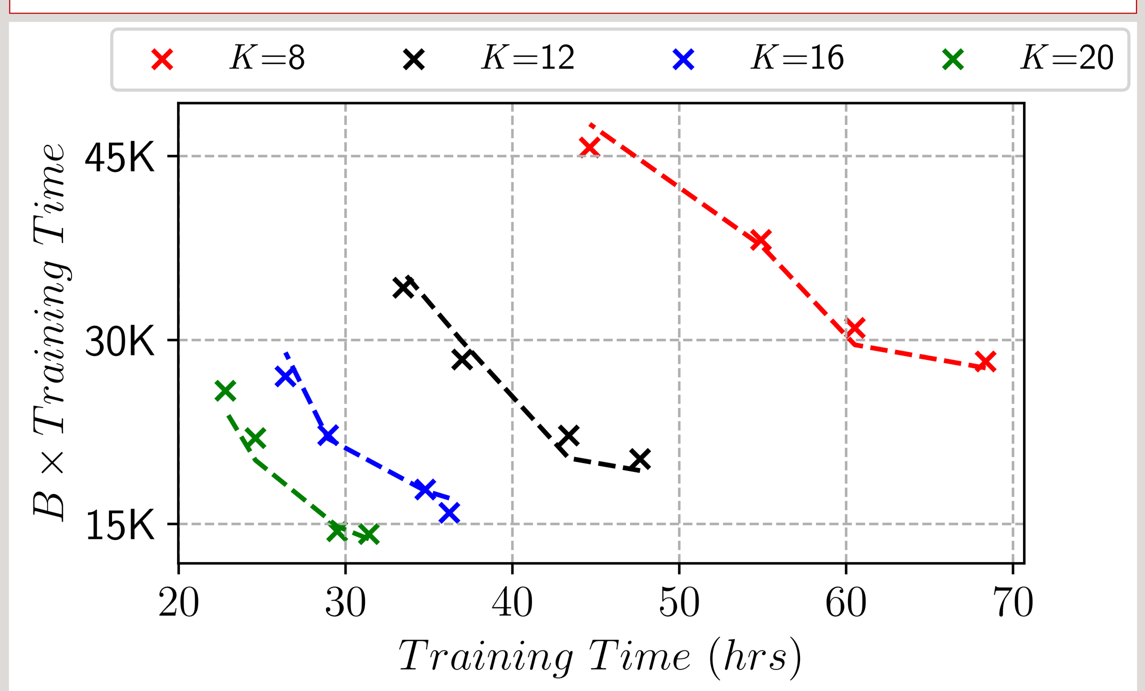
- Due to Amdahl's law, training does not scale linearly. Adding 4x resources does not reduce time by 4x!



Evaluation

- Cost-Time tradeoff to converge for ResNet18, ResNet50 and Transformer
- VMs priced by memory allocated; For each K, models evaluated for B (384,512,768,1024)
- **Scatter points are real time + cost of different (K,B) configurations**
- **Dashed line is performance predicted by Scavenger**
- **From the predicted and actual performance, tradeoff between time and cost exists and detected by Scavenger!**

ResNet50



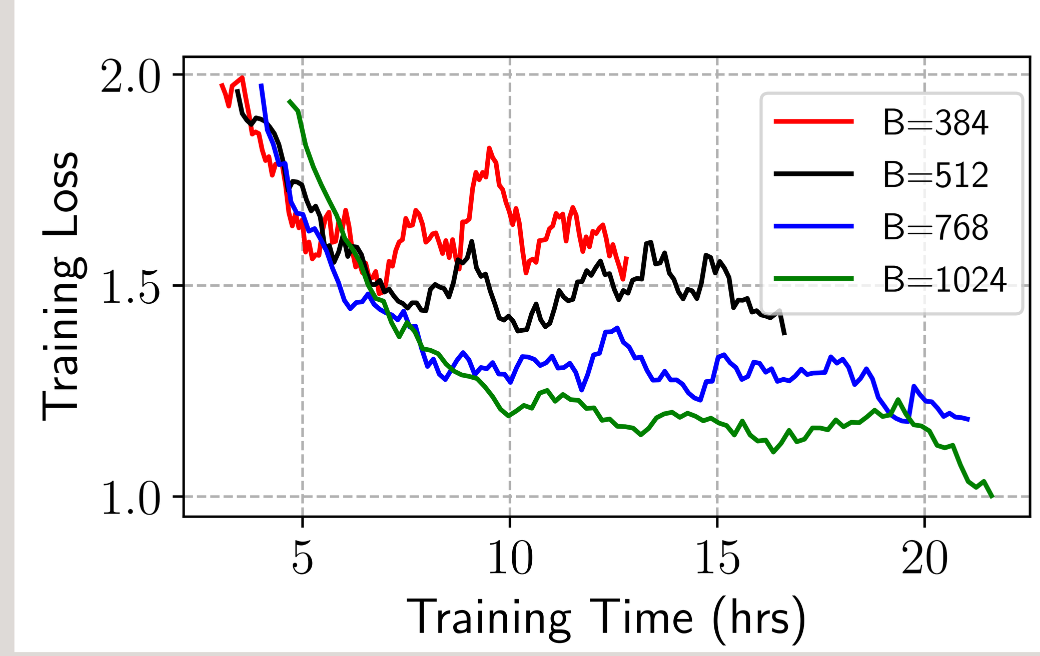
Research Questions

- How effective is gradient noise as an indicator of statistical efficiency?
- How accurate is our performance and cost model across different job configurations?
- What are the performance and cost tradeoffs for different cost models in the cloud?
- What savings can be achieved with our job configurations & resource allocation policies?

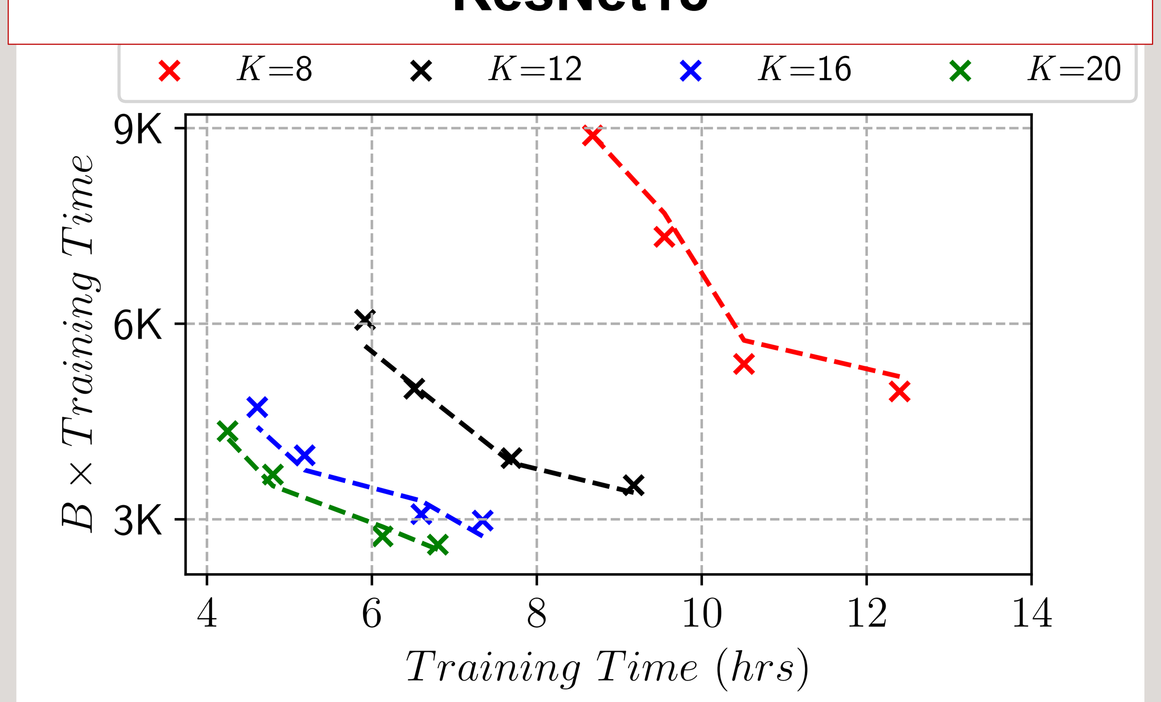
Vertical scaling in the cloud

- Training with large batches is efficient and reduces time to training loss but increases memory utilization!

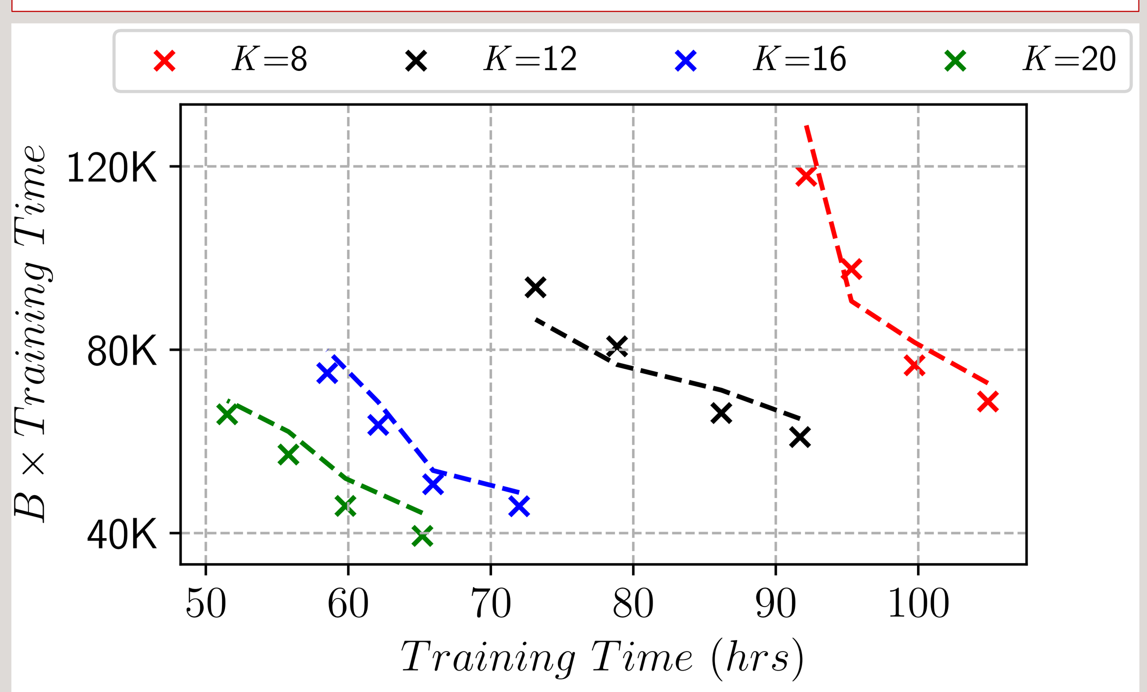
ResNet18



ResNet18



Transformer



Conclusion:

- Scavenger uses online profiling and new parallel and statistical performance models for estimating the training performance on different cloud configurations, **with high accuracy of over 98%, and reduces training time by 2×.**