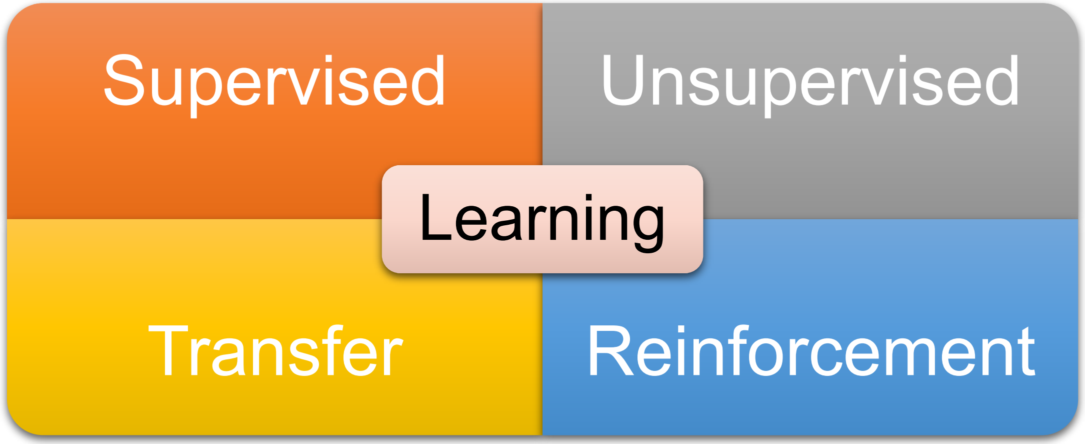
SLAQ: Quality-Driven Scheduling for Distributed Machine Learning

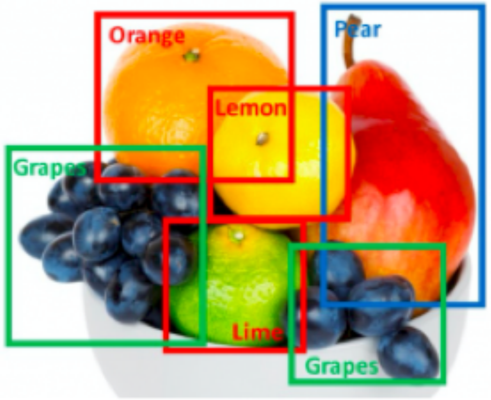
Haoyu Zhang*, Logan Stafman*, Andrew Or, Michael J. Freedman

PRINCETON UNIVERSITY

“AI is the new electricity.”

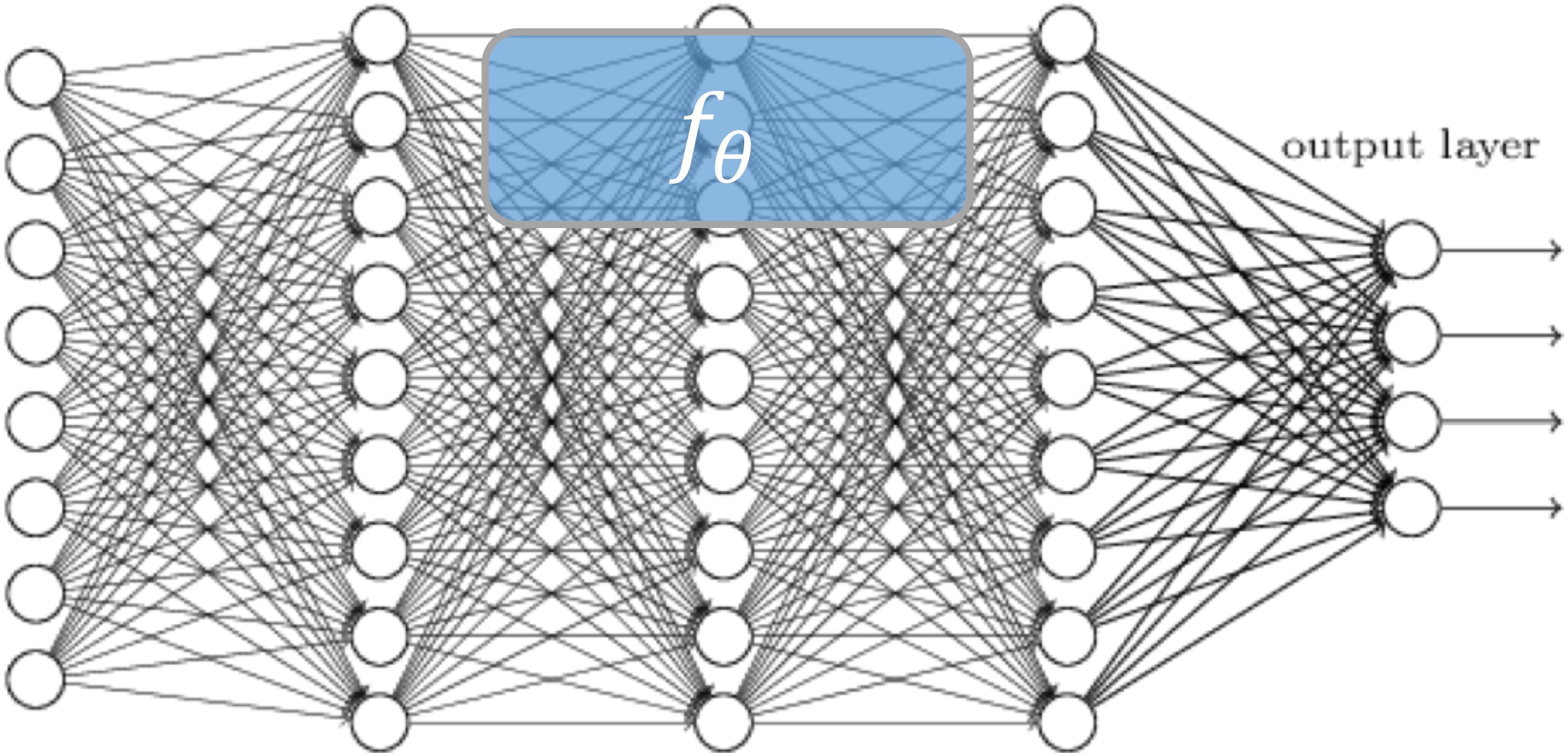
- Machine translation
- Recommendation system
- Autonomous driving
- Object detection and recognition





ML algorithms are *approximate*

- ML model: a parametric transformation



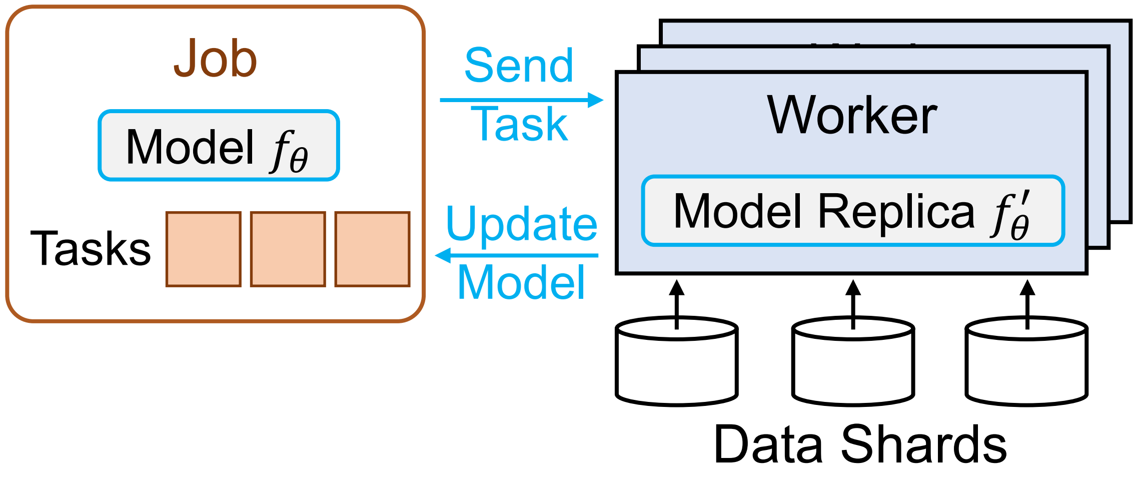
ML algorithms are *approximate*

- ML model: a parametric transformation

  - maps input variables X to output variables Y
  - typically contains a set of parameters θ

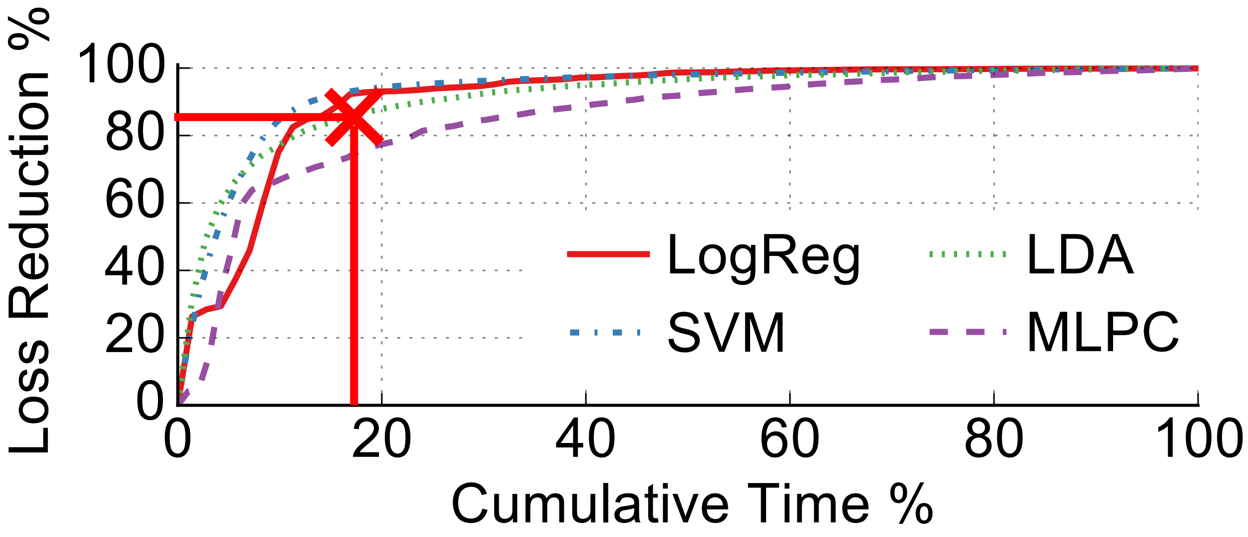
- Quality: how well model maps input to the correct output
- Loss function: discrepancy of model output and ground truth

Training ML models: an *iterative* process



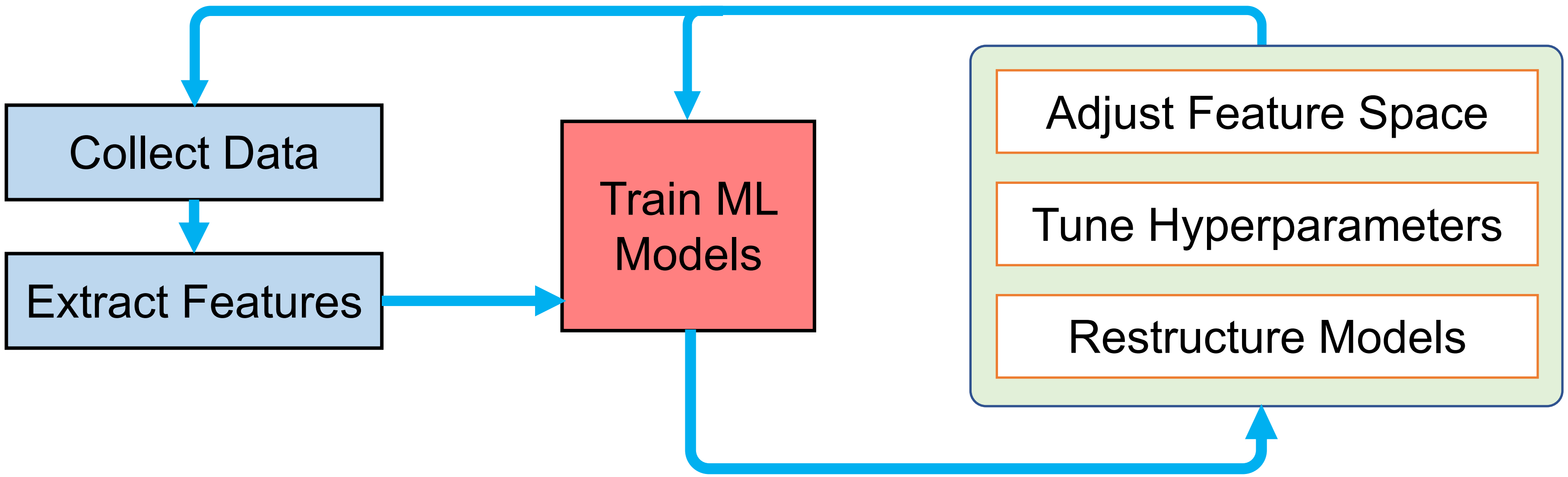
- Training algorithms iteratively minimize a loss function
  - E.g., stochastic gradient descent (SGD), L-BFGS

Training ML models: an *iterative* process



- Quality improvement is subject to diminishing returns
  - More than 80% of work done in 20% of time

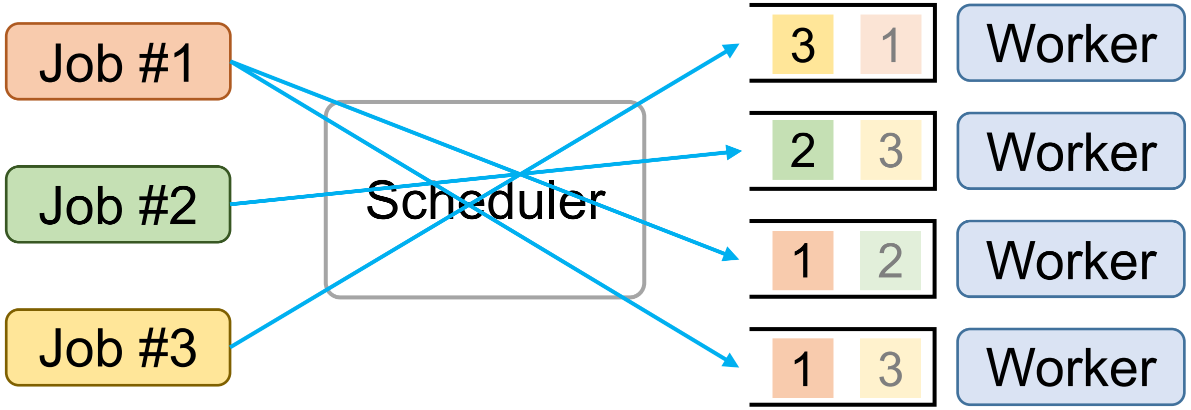
Exploratory ML training: not a one-time effort



- Train model multiple times for exploratory purposes
- Provide early feedback, direct model search for high quality models

How to schedule multiple training jobs on shared cluster?

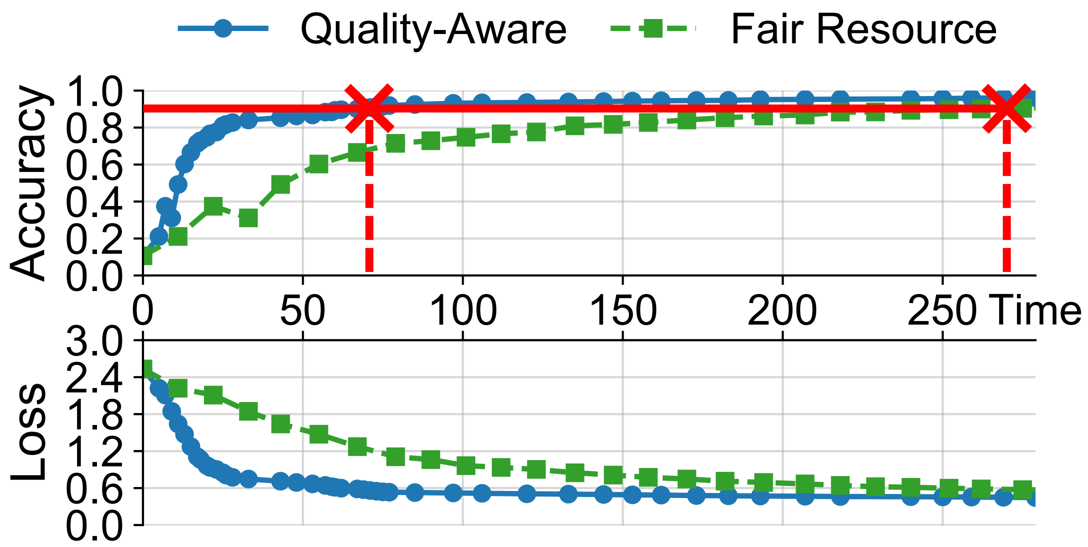
- Key features of ML jobs
  - Approximate
  - Diminishing returns
  - Exploratory process



- Problem with resource fairness scheduling
  - Jobs in early stage: could benefit a lot from additional resources
  - Jobs almost converged: make only marginal improvement

SLAQ: quality-aware scheduling

- Intuition: in the context of approximate ML training, more resources should be allocated to jobs that have the most potential for quality improvement



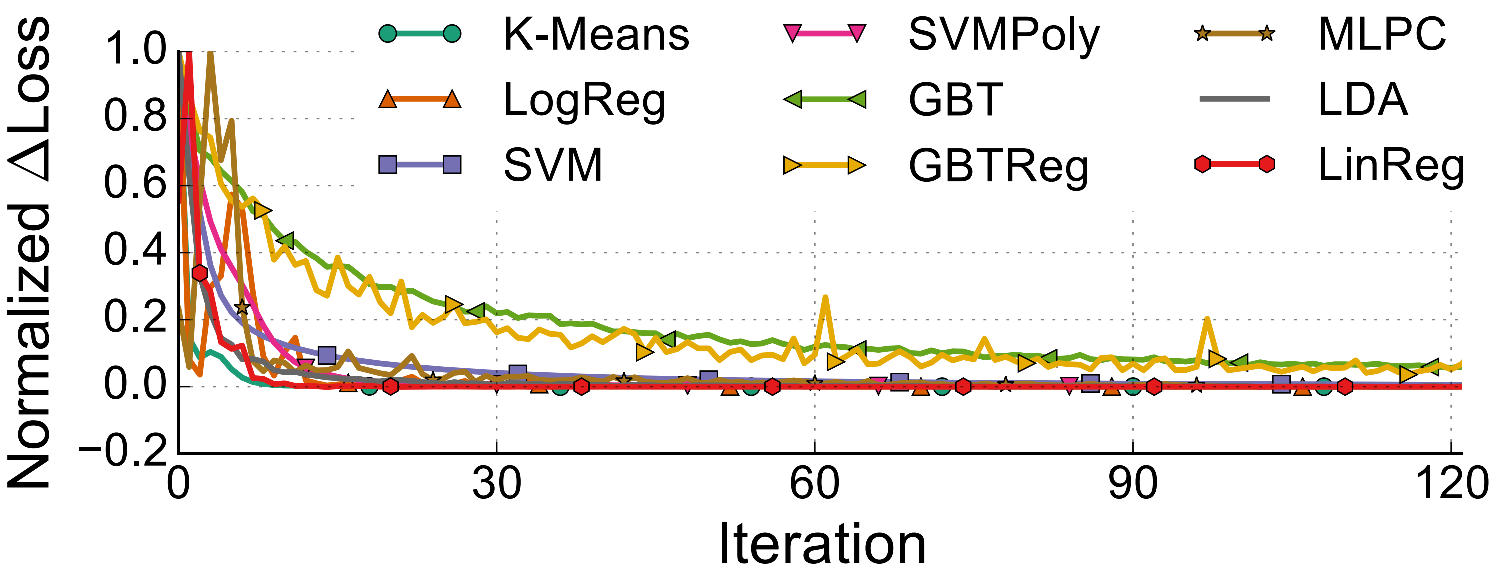
Solution Overview

Normalizing quality metrics

	Applicable to All Algorithms?	Comparable Magnitudes?	Known Range?	Predictable?
Accuracy / F1 Score / Area Under Curve / Confusion Matrix / etc.	✗	✓	✓	✗
Loss	✓	✗	✗	✓
Normalized Loss	✓	✓	✗	✓
ΔLoss	✓	✗	✓	✓
Normalized ΔLoss	✓	✓	✓	✓

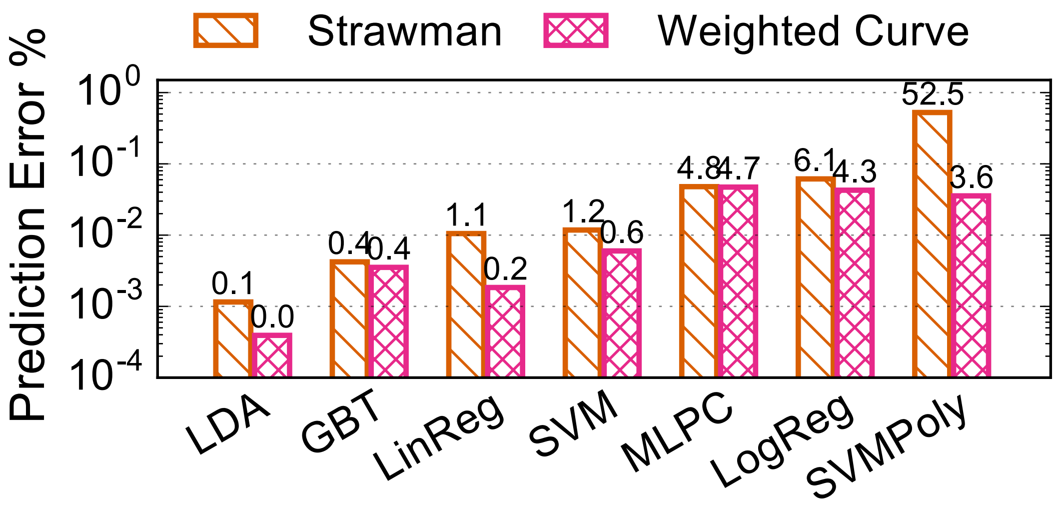
Normalizing quality metrics

- Normalize change of loss values *w.r.t.* largest change so far
  - Currently does not support some non-convex optimization algorithms



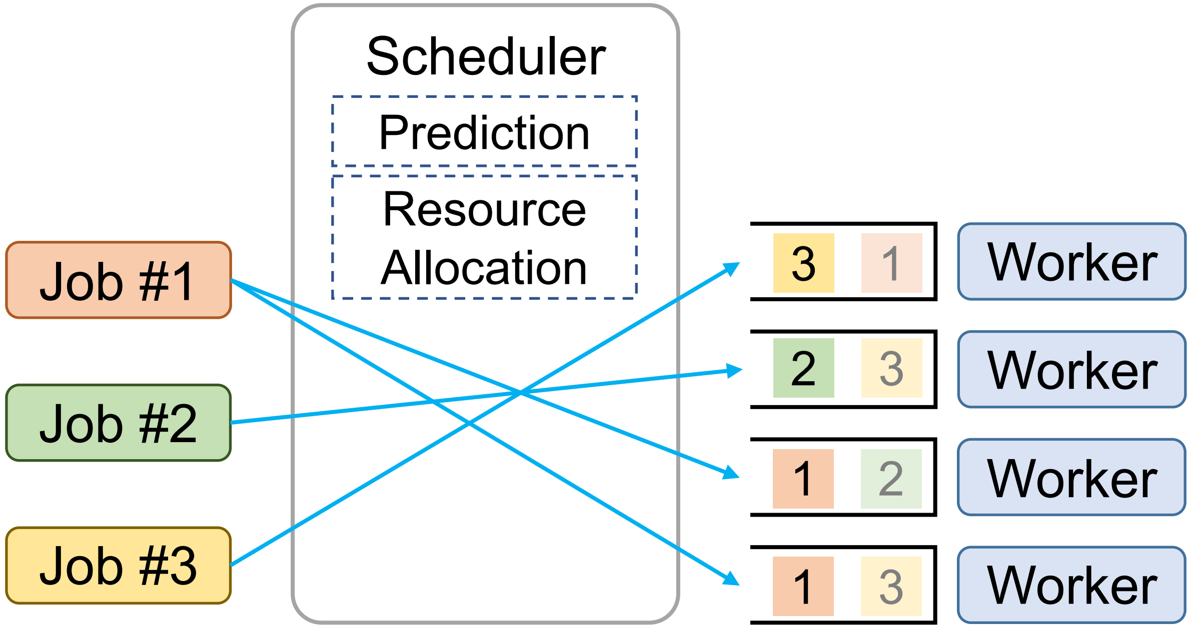
Training iterations: loss prediction

- Previous work: offline profiling / analysis [Ernest NSDI 16] [CherryPick NSDI 17]
    - Overhead for frequent offline analysis is huge
- Strawman: use last ΔLoss as prediction for future ΔLoss
- SLAQ: online prediction using weighted curve fitting



Scheduling approximate ML training jobs

- Predict how much quality can be improved when assign X workers to jobs
- Reassign workers to maximize quality improvement



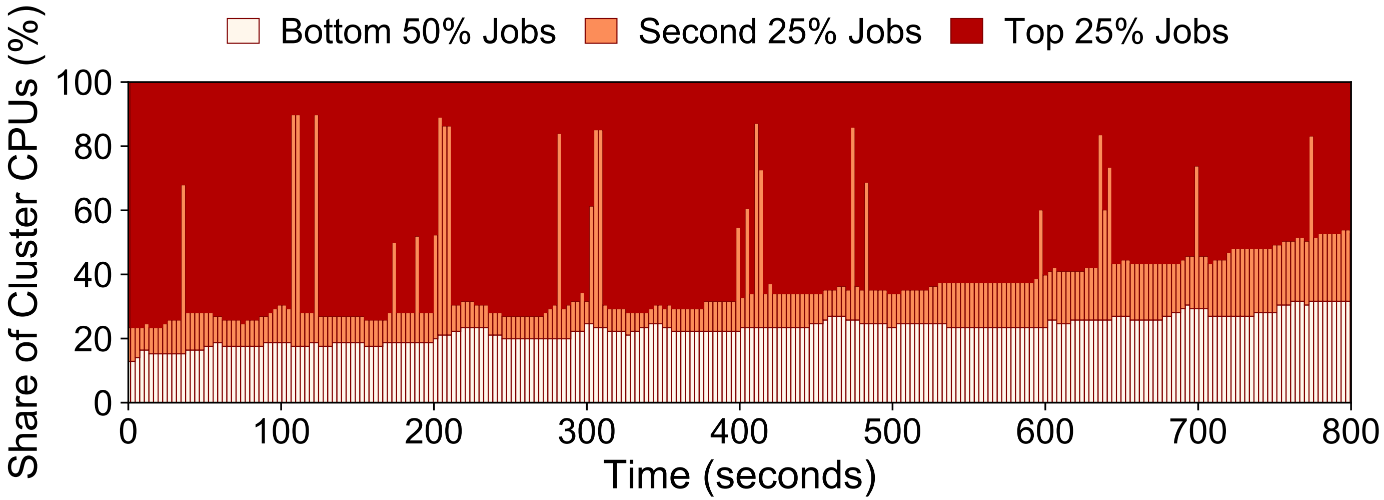
Experiment setup

- Representative mix of training jobs with Spark MLlib
- Compare against a work-conserving fair scheduler

Algorithm	Acronym	Type	Optimization Algorithm	Dataset
K-Means	K-Means	Clustering	Lloyd Algorithm	Synthetic
Logistic Regression	LogReg	Classification	Gradient Descent	Epsilon [33]
Support Vector Machine	SVM	Classification	Gradient Descent	Epsilon
SVM (polynomial kernel)	SVMPoly	Classification	Gradient Descent	MNIST [34]
Gradient Boosted Tree	GBT	Classification	Gradient Boosting	Epsilon
GBT Regression	GBTReg	Regression	Gradient Boosting	YearPredictionMSD [35]
Multi-Layer Perceptron Classifier	MLPC	Classification	L-BFGS	Epsilon
Latent Dirichlet Allocation	LDA	Clustering	EM / Online Algorithm	Associated Press Corpus [36]
Linear Regression	LinReg	Regression	L-BFGS	YearPredictionMSD

Evaluation: resource allocation across jobs

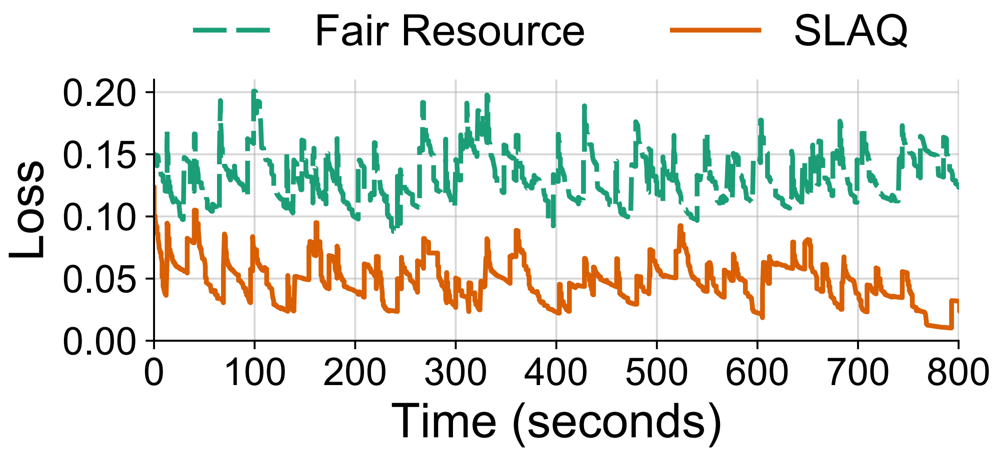
- 160 training jobs submitted to cluster following Poisson distribution
  - 25% jobs with high loss values
  - 25% jobs with medium loss values
  - 50% jobs with low loss values (almost converged)



Evaluation: cluster-wide quality and time

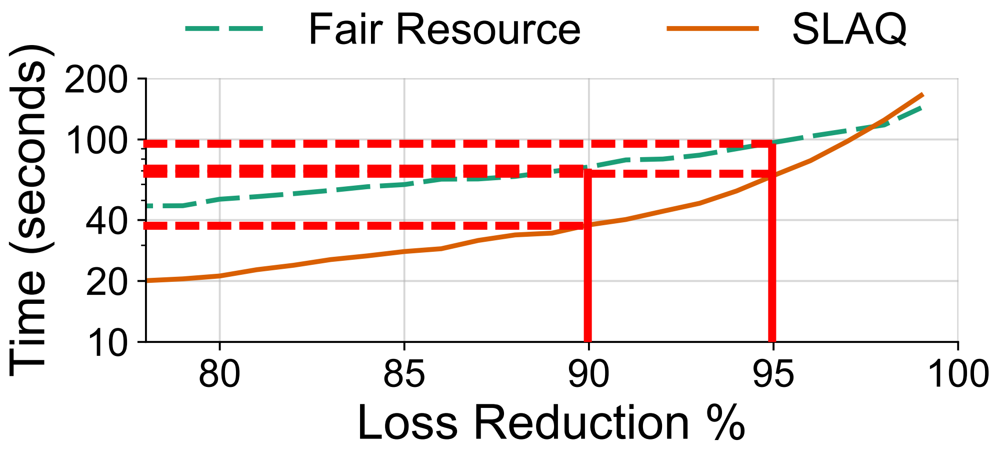
Quality

- SLAQ's average loss is 73% lower than that of the fair scheduler



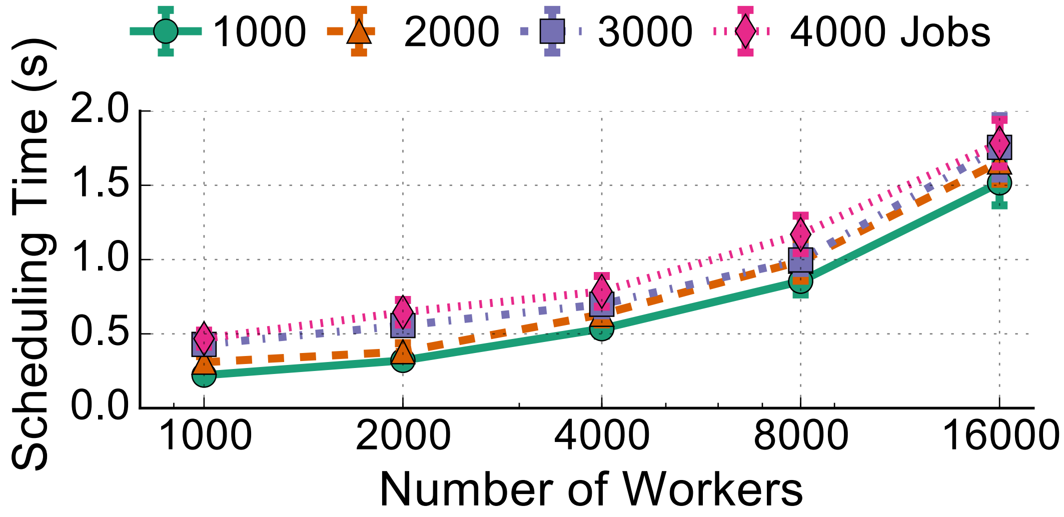
Time

- SLAQ reduces time to reach 90% (95%) loss reduction by 45% (30%)



SLAQ Evaluation: Scalability

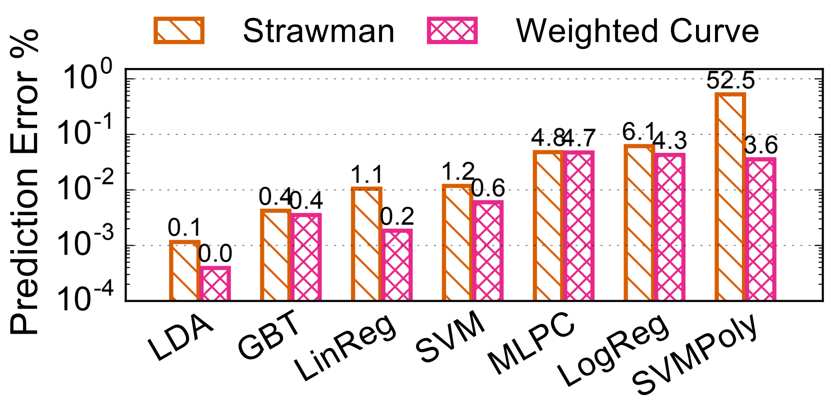
- Frequently reschedule and reconfigure in reaction to changes of progress
- Even with thousands of concurrent jobs, SLAQ makes rescheduling decisions in just a few seconds

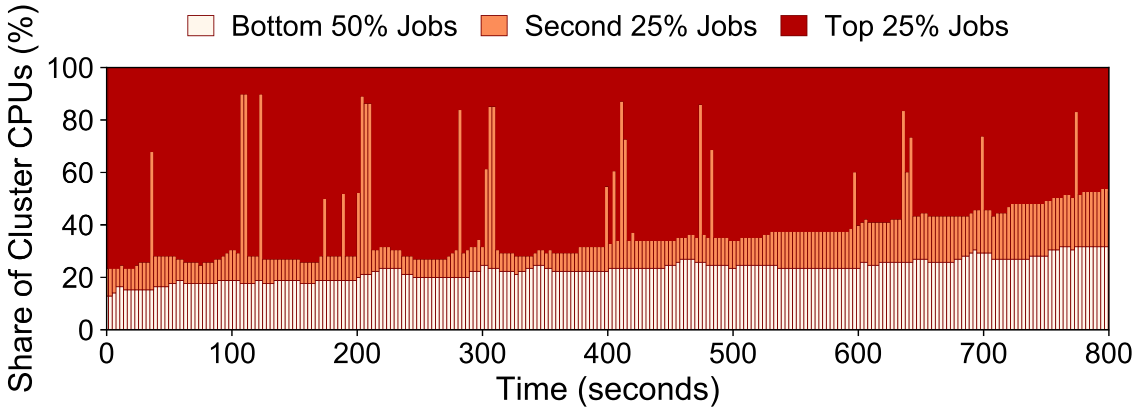


Conclusion

- SLAQ leverages the approximate and iterative ML training process

- Highly tailored prediction for iterative job quality
- Allocate resources to maximize quality improvement





- SLAQ achieves better overall quality and end-to-end training time

Training iterations: runtime prediction

- Iteration runtime: $c \cdot S/N$
  - Model complexity c, data size S, number of workers N
  - Model update (i.e., size of $\Delta\theta$) is comparably much smaller