End-to-End Inventory Prediction and Contract Allocation for Guaranteed Delivery Advertising

Wuyang Mao[1], Chuanren Liu[2*], Yundu Huang[1], Zhonglin Zu[1], M Harshvardhan[2], Liang Wang[1], Bo Zheng[1]

[1]Alibaba Group, China; [2]The University of Tennessee, Knoxville

29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining | Long Beach, CA | August 5-10, 2023

Background

Guaranteed Delivery (GD) Advertising is a strategic approach where advertisers secure their desired inventory of advertising impressions in advance by signing contracts with publishers weeks or months ahead of the targeting dates

GD Advertising places a fix price on impressions adding stability

It had better controls on allocation and targeting audience

Real-time Bidding (RTB) is dynamic and competitive, which advertisers don't like as much

RTB can be more computationally expensive

Allocation decision comes down to a bipartitie graph problem where supply of users has to be matched to the demand from the advertisers for the impressions. This results in an elegant mathematical problem of prediction and optimization.

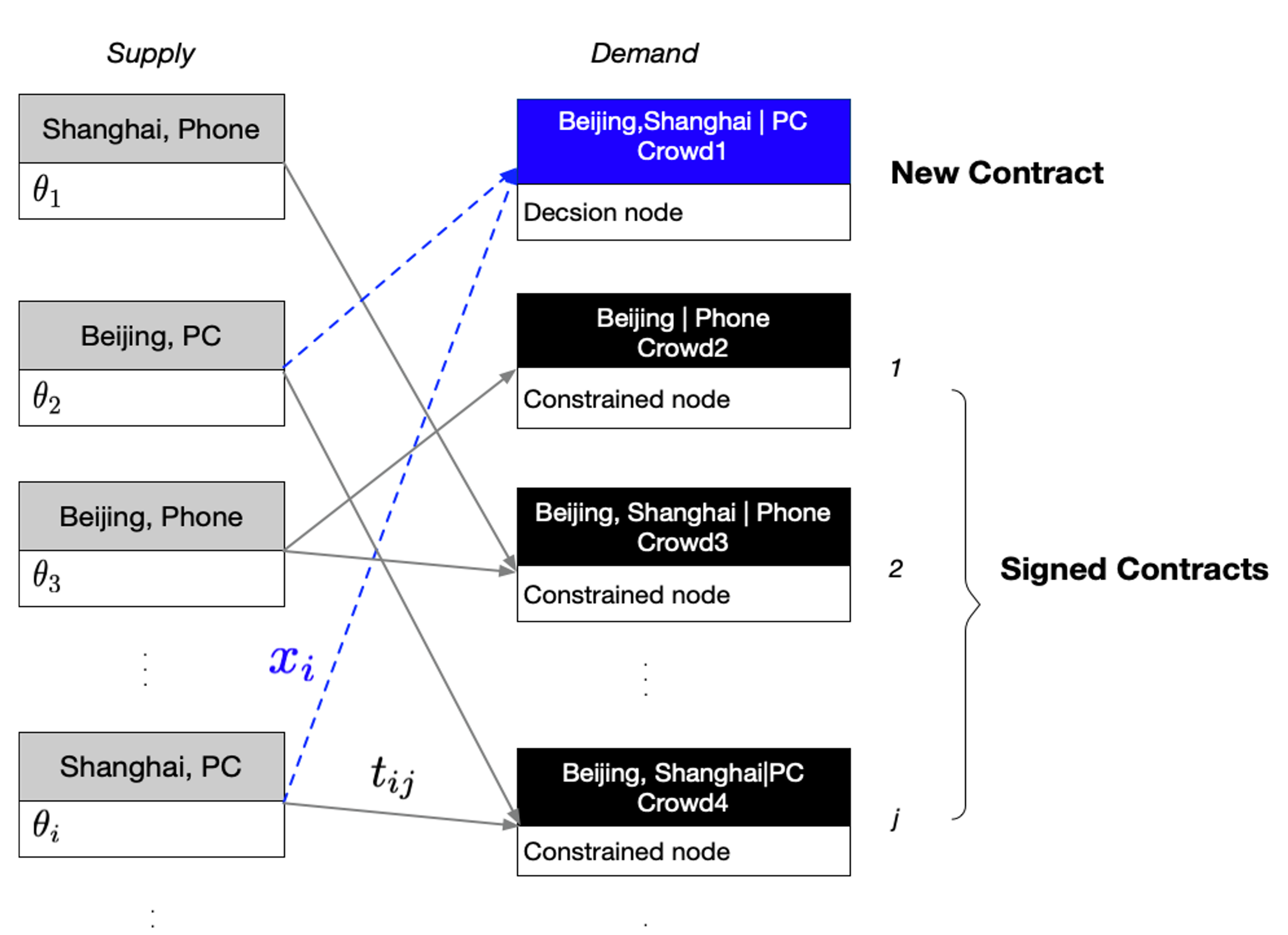
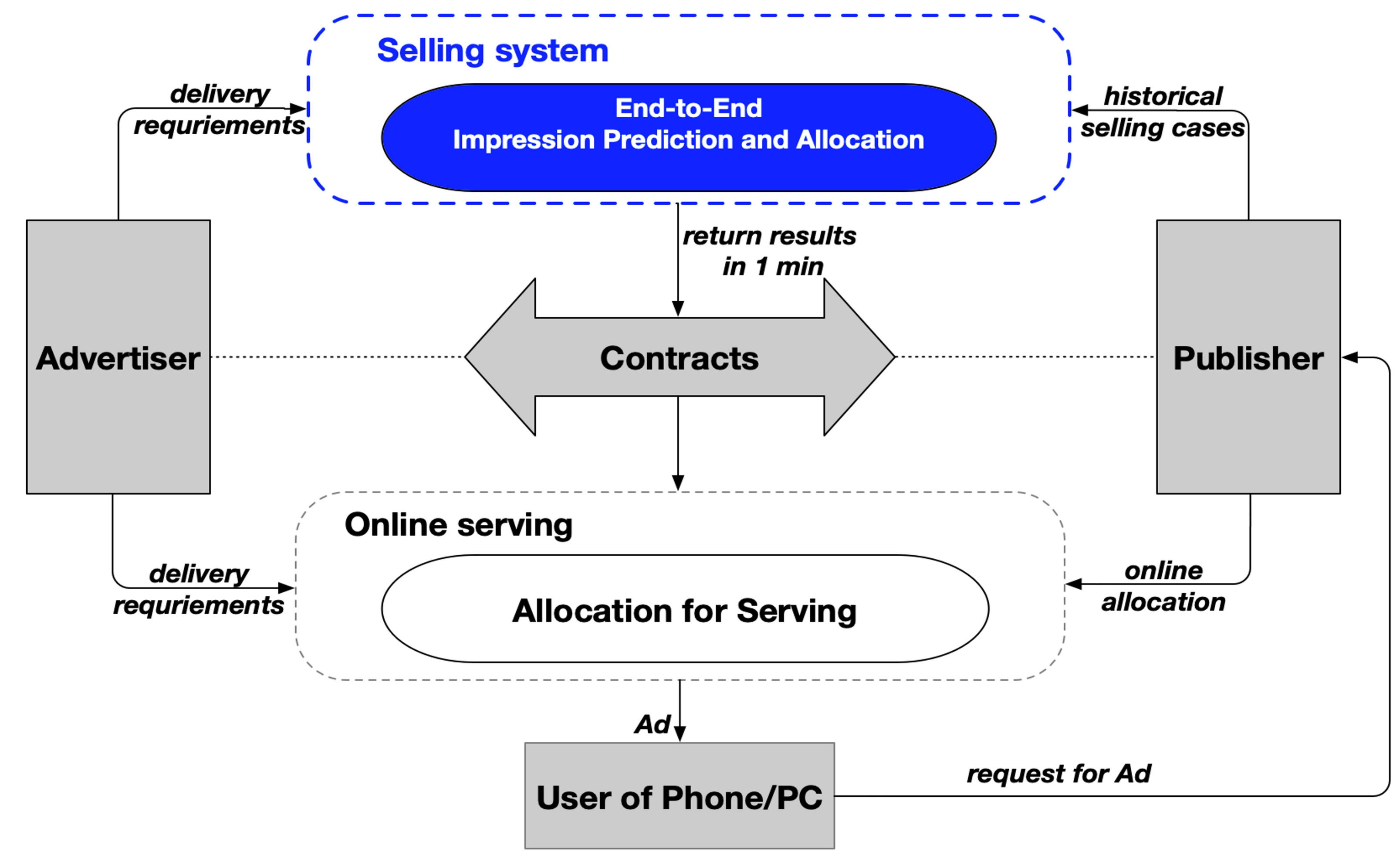


Figure: Bipartite graph of contract allocation problem. Supply nodes are impression inventories while demand nodes are impression contracts

System Architecture

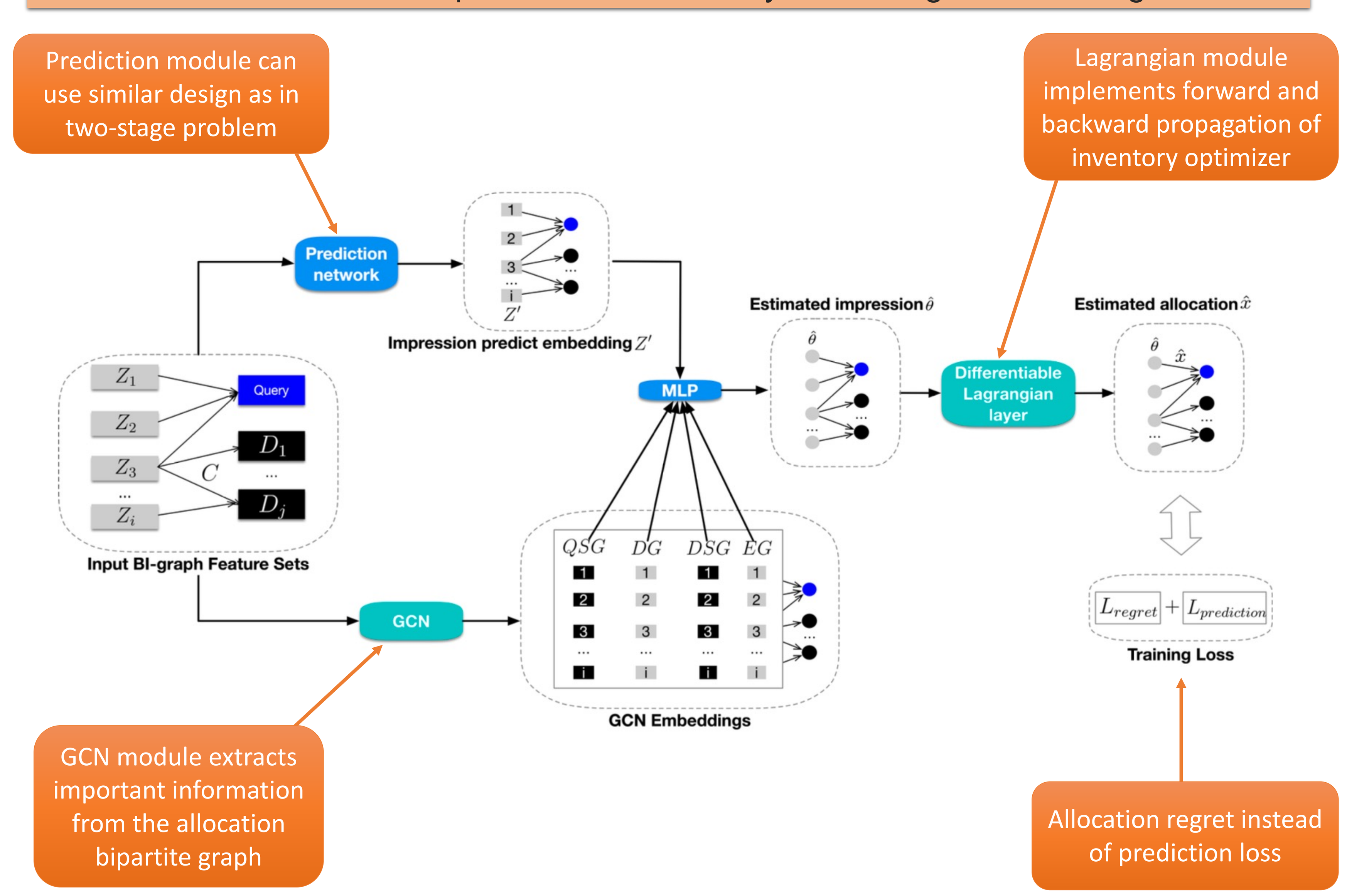
System Architecture for Guaranteed Delivery (GD) Advertisement at Alibaba's websites where advertisers and publishers enter into a contract for impressions and have delivery requirements



Neural Langrangian Selling (NLS)

Traditionally, the problem has been solved in two stages:
1. Predicting the impressions from each user base using some predictive model
2. Optimizing allocation based on the predicted impressions using linear programming

Our proposed solution (**Neural Langrangian Selling**) blends the two stages into a single End-to-End Prediction and Optimization solution by minimizing allocation regret



Key Innovations

Graph Convolutional Networks (GCN) create embeddings responsible for constraints in our models

Differentiable Lagrangian Layer backpropagates allocation error to the neural network creating an end-to-end model

Results

Benchmark Models

- **Two-stage Model**
  - Compares end-to-end approach with traditional two-stage method
- **Pure Fully-Connected (PF)**
  - Baseline end-to-end approach using a simple black-box neural network
- **Pure Prediction GCN (PPG)**
  - Removes Lagrangian layer from NLS, uses GCN for prediction
- **Prediction Network + Lagrangian solver (PL)**
  - End-to-end approach without GCN module
- **GCN+QPTL/GCN+InOpt (End-to-End)**
  - LP solvers IntOpt and QPTL to compare with Lagrangian layer

Evaluation Metrics

- End-to-End Normalized Deviation Error
- First-stage Error
- Second-stage Error
- Publisher Revenue (Avg. Revenue/Day)
- Delivery Rate (Delivered/Promised)
- Usage Rate (Sold/Available Impressions)

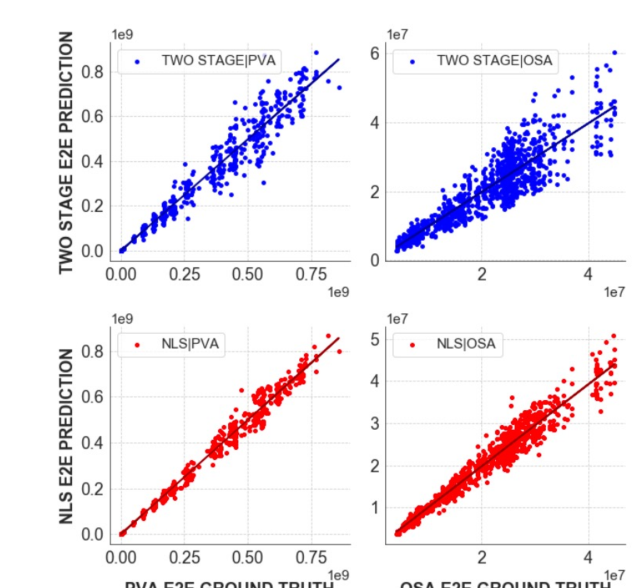
NLS outperforms other models on most benchmarks

Experiment results on **Offline data**

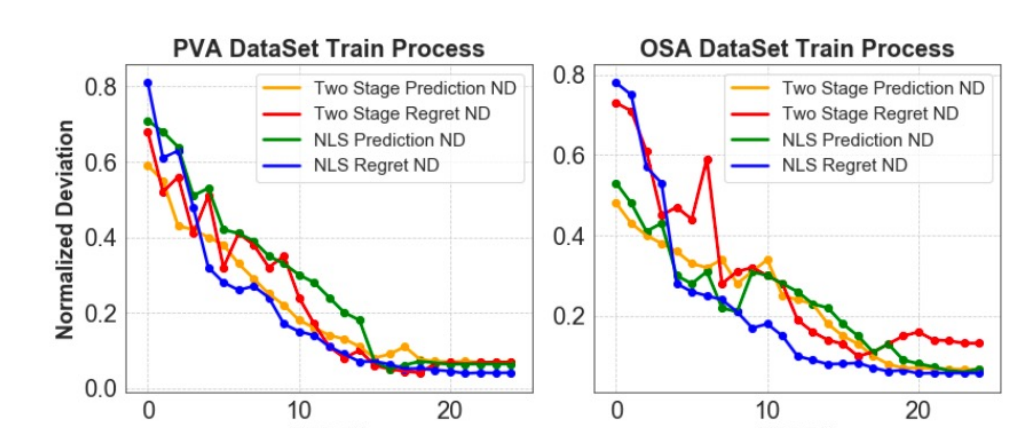
Methods	full targeting		single targeting		random targeting	
	ND_{pre}	ND_{reg}	ND_{pre}	ND_{reg}	ND_{pre}	ND_{reg}
Two Stage	$0.101_{\pm 0.003}$	$0.023_{\pm 2e-4}$	$0.101_{\pm 0.003}$	$0.125_{\pm 0.005}$	$0.101_{\pm 0.003}$	$0.045_{\pm 0.002}$
PF	$0.130_{\pm 0.010}$	$0.045_{\pm 0.005}$	$0.115_{\pm 0.008}$	$0.132_{\pm 0.010}$	$0.121_{\pm 0.010}$	$0.076_{\pm 0.007}$
PPG	$0.125_{\pm 0.010}$	$0.015_{\pm 1e-4}$	$0.127_{\pm 0.007}$	$0.112_{\pm 0.001}$	$0.135_{\pm 0.011}$	$0.036_{\pm 0.001}$
PL	$0.102_{\pm 0.001}$	$0.008_{\pm 1e-4}$	$0.103_{\pm 0.001}$	$0.113_{\pm 0.003}$	$0.101_{\pm 0.002}$	$0.047_{\pm 2e-4}$
NLS	$\mathbf{0.096}_{\pm 0.002}$	$\mathbf{0.007}_{\pm 2e-4}$	$\mathbf{0.097}_{\pm 0.001}$	$\mathbf{0.098}_{\pm 0.001}$	$\mathbf{0.095}_{\pm 0.001}$	$\mathbf{0.029}_{\pm 1e-4}$

Experiment results on **Online data**

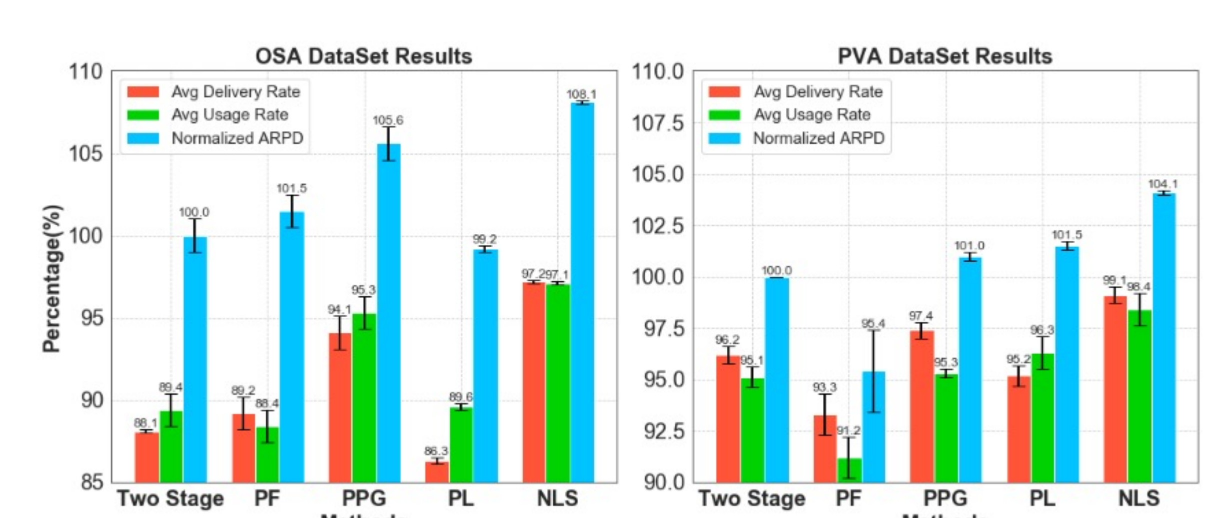
Methods	PVA		OSA	
	ND_{pre}	ND_{reg}	ND_{pre}	ND_{reg}
Two Stage	$0.069_{\pm 0.002}$	$0.068_{\pm 0.004}$	$0.067_{\pm 0.002}$	$0.132_{\pm 0.005}$
PF	$0.083_{\pm 0.015}$	$0.095_{\pm 0.020}$	$0.075_{\pm 0.015}$	$0.128_{\pm 0.021}$
PPG	$0.085_{\pm 0.005}$	$0.054_{\pm 0.002}$	$0.078_{\pm 0.003}$	$0.086_{\pm 0.004}$
PL	$0.065_{\pm 0.002}$	$0.061_{\pm 0.002}$	$\mathbf{0.065}_{\pm 0.001}$	$0.136_{\pm 0.004}$
NLS	$\mathbf{0.064}_{\pm 0.003}$	$\mathbf{0.041}_{\pm 0.001}$	$0.068_{\pm 0.003}$	$\mathbf{0.058}_{\pm 0.001}$



NLS has fewer outliers in comparison with two-stage methods



NLS has more stable training trajectory



NLS has better delivery rate, usage rate and publisher revenue